Online Platt Scaling with Calibeating

Chirag Gupta [1] **Aaditya Ramdas** [1]

Abstract

We present an online post-hoc calibration method, called Online Platt Scaling (OPS), which combines the Platt scaling technique with online logistic regression. We demonstrate that OPS smoothly adapts between i.i.d. and non-i.i.d. settings with distribution drift. Further, in scenarios where the best Platt scaling model is itself miscalibrated, we enhance OPS by incorporating a recently developed technique called calibeating to make it more robust. Theoretically, our resulting OPS+calibeating method is guaranteed to be calibrated for adversarial outcome sequences. Empirically, it is effective on a range of synthetic and real-world datasets, with and without distribution drifts, achieving superior performance without hyperparameter tuning. Finally, we extend all OPS ideas to the beta scaling method.

1 Introduction

In the past two decades, there has been significant interest in the ML community on post-hoc calibration of ML classifiers (Zadrozny and Elkan, 2002; Niculescu-Mizil and Caruana, 2005; Guo et al., 2017). Consider a pretrained classifier $f : \mathcal{X} \to [0, 1]$ that produces scores in $[0, 1]$ for covariates in $\mathcal{X}$. Suppose f is used to make probabilistic predictions for a sequence of points $(\mathbf{x}_t, y_t)_{t=1}^T$ where $y_t \in \{0, 1\}$. Informally, f is said to be calibrated (Dawid, 1982) if the predictions made by f match the empirically observed frequencies when those predictions are made:

$$\text{for all } p \in [0, 1], \text{Average}\{y_t : f(\mathbf{x}_t) \approx p\} \approx p. \quad (1)$$

In practice, for well-trained f, larger scores $f(\mathbf{x})$ indicate higher likelihoods of $y = 1$, so that f does well for accuracy or a ranking score like AUROC. Yet we often find that f does not satisfy (some formalized version of) condition (1). The goal of post-hoc calibration, or recalibration, is to use *held-out data* to learn a low-complexity mapping $m : [0, 1] \to [0, 1]$ so that $m(f(\cdot))$ retains the good properties of f—accuracy, AUROC, sharpness—as much as possible, but is better calibrated than f.

The focus of this paper is on a recalibration method proposed by Platt (1999), commonly known as Platt scaling (PS). The PS mapping m is a sigmoid transform over f parameterized by two scalars $(a, b) \in \mathbb{R}^2$:

$$m^{a,b}(f(\mathbf{x})) := \text{sigmoid}(a \cdot \text{logit}(f(\mathbf{x})) + b). \quad (2)$$

Here $\text{logit}(z) = \log(z/1-z)$ and $\text{sigmoid}(z) = 1/1+e^{-z}$ are inverses. Thus $m^{1,0}$ is the identity mapping. Figure 1a has additional illustrative $m^{a,b}$ plots; these are easily interpreted—if f is overconfident, that is if $f(x)$ values are skewed towards 0 or 1, we can pick $a \in (0, 1)$ to improve calibration; if f is underconfident, we can pick $a > 1$; if f is systematically biased towards 0 (or 1), we can pick $b > 0$ (or $b < 0$). The counter-intuitive choice $a < 0$ can also make sense if f's predictions oppose reality (perhaps due to a distribution shift). The parameters (a, b) are typically set as those that minimize log-loss over calibration data or equivalently maximize log-likelihood under the model $y_i \overset{\text{iid}}{\sim} \text{Bernoulli}(m^{a,b}(f(\mathbf{x}_i)))$, on the held-out data.

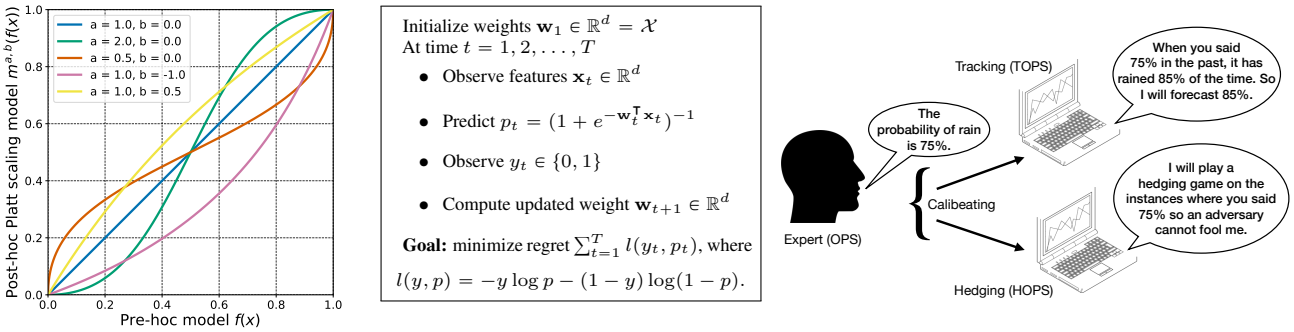
Although a myriad of recalibration methods now exist, PS remains an empirically strong baseline. In particular, PS is effective when few samples are available for recalibration (Niculescu-Mizil and Caruana, 2005; Gupta and Ramdas, 2021). Scaling before subsequent binning has emerged as a useful methodology (Kumar et al., 2019; Zhang et al., 2020). Multiclass adaptations of PS, called temperature, vector, and matrix scaling have become popular (Guo et al., 2017). Being a parametric method, however, PS comprises a limited family of post-hoc corrections—for instance, since $m^{a,b}$ is always a monotonic transform, PS must fail even for i.i.d. data for some data-generating distributions (see Gupta et al. (2020) for a formal proof). Furthermore, we are interested in going beyond i.i.d. data to data with drifting/shifting distribution. This brings us to our first question,

(Q1) Can PS be extended to handle shifting or drifting data distributions?

A separate view of calibration that pre-dates the ML post-hoc calibration literature is the online adversarial calibration

[1]Carnegie Mellon University, Pittsburgh PA, USA. Correspondence to: Chirag Gupta <chiragg@cmu.edu, chiragpvg@gmail.com>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023.



(a) Platt scaling (b) Online logistic regression (c) Calibeating applied to online Platt scaling

Figure 1: The combination of Platt scaling and online logistic regression yields Online Platt Scaling (OPS). Calibeating is applied on top of OPS to achieve further empirical improvements and theoretical validity.

framework (DeGroot and Fienberg, 1981; Foster and Vohra, 1998). Through the latter work, we know that calibration can be achieved for arbitrary y_t sequences without relying on a pretrained model f or doing any other modeling over available features. This is achieved by hedging or randomizing over multiple probabilities, so that "the past track record can essentially only improve, no matter the future outcome" (paraphrased from Foster and Hart (2021)). For interesting classification problems, however, the y_t sequence is far from adversarial and informative covariates $\mathbf{x}_t$ are available. In such settings, covariate-agnostic algorithms achieve calibration by predicting something akin to an average $\sum_{s=1}^t y_s/t$ at time $t + 1$ (see Appendix D). Such a prediction, while calibrated, is arguably not useful. A natural question is:

(Q2) Can informative covariates (features) be used to make online adversarial calibration practical?

We find an answer to (Q1) and (Q2) by leveraging the recently developed framework of calibeating (Foster and Hart, 2023), which is illustrated in Figure 1c. Calibeating shows how to perform certain *corrections* on top of pre-existing *expert* forecasts to improve (*calibeat*) them. The key calibeating idea relevant to our study is that adversarial calibration techniques can be used to simultaneously beat an expert *and* be provably calibrated.

For the sequential problem of being calibrated for a stream $(\mathbf{x}_t, y_t)_{t\geqslant 1}$, where is one to find an expert forecaster that is both covariate-based and time-adaptive? One possibility is a time-series model—a sequence of predictors $(f_t : \mathcal{X} \to [0, 1])_{t\geqslant 1}$ that are updated as more data becomes available, potentially using autoregression. Our forthcoming proposal is simpler and arguably requires less (domain) expertise.

1.1 Online adversarial post-hoc calibration

The proposal, summarized in Figure 2, is as follows. First, train any probabilistic classifier f on some part of the data. Then, perform *online post-hoc calibration* on top of f to get

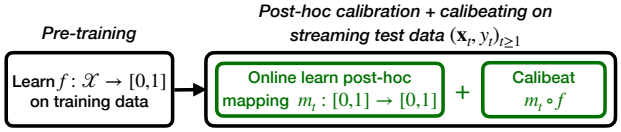
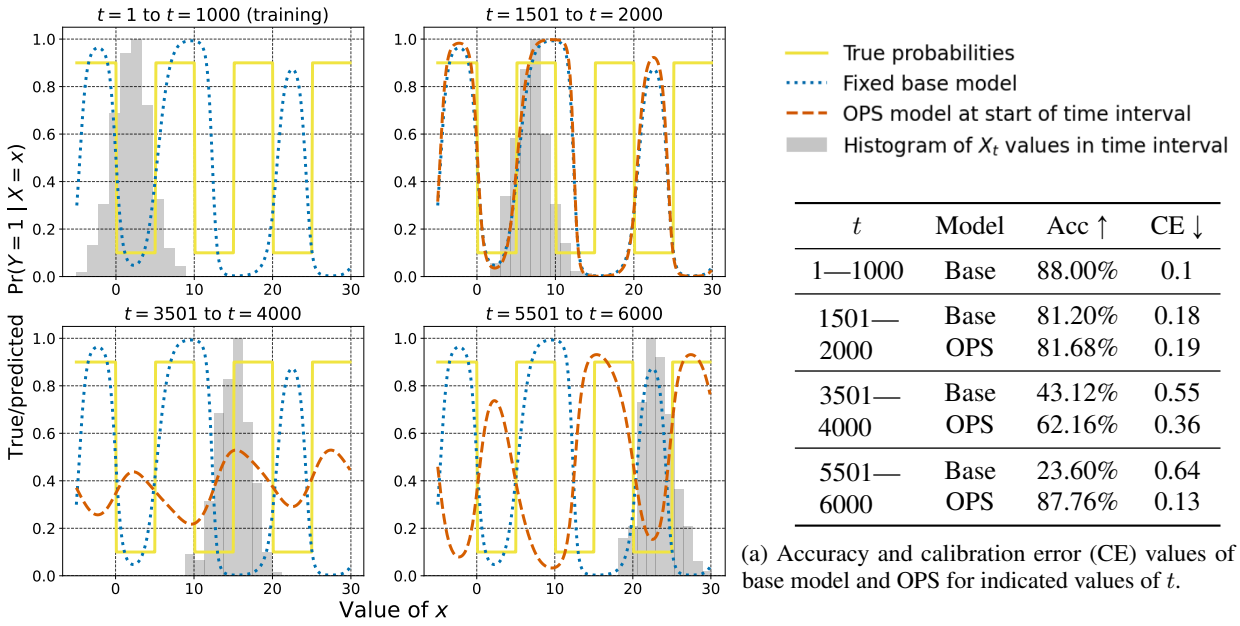


Figure 2: Online adversarial post-hoc calibration.

online adaptivity. In effect, this amounts to viewing $f(\mathbf{x}_t)$ as a scalar "summary" of $\mathbf{x}_t$, and the post-hoc mapping $(m_t : [0, 1] \to [0, 1])_{t\geqslant 1}$ becomes the time-series model over the scalar feature $f(\mathbf{x}_t)$. Finally, apply calibeating on the post-hoc predictions $m_t(f(\mathbf{x}_t))$ to obtain adversarial validity. Figure 2 highlights our choice to do both post-hoc calibration and calibeating simultaneously on the streaming test data $(\mathbf{x}_t, y_t)_{t\geqslant 1}$.

Such an online version of post-hoc calibration has not been previously studied to the best of our knowledge. We show how one would make PS online, to obtain Online Platt Scaling (OPS). OPS relies on a simple but crucial observation: PS is a two-dimensional logistic regression problem over "pseudo-features" $\text{logit}(f(\mathbf{x}_t))$. Thus the problem of learning OPS parameters is the problem of online logistic regression (OLR, see Figure 1b for a brief description). Several regret minimization algorithms have been developed for OLR (Hazan et al., 2007; Foster et al., 2018; Jézéquel et al., 2020). We consider these and find an algorithm with optimal regret guarantees that runs in linear time. These regret guarantees imply that OPS is guaranteed to perform as well as the best fixed PS model in hindsight for an arbitrarily distributed online stream $(\mathbf{x}_t, y_t)_{t\geqslant 1}$, which includes the entire range of distribution drifts—i.i.d. data, data with covariate/label drift, and adversarial data. We next present illustrative experiments where this theory bears out impressively in practice.

Then, Section 2 presents OPS, Section 3 discusses calibeating, Section 4 presents baseline experiments on synthetic and real-world datasets. Section 5 discusses the extension of all OPS ideas to a post-hoc technique called beta scaling.



t	Model	Acc ↑	CE ↓
1—1000	Base	88.00%	0.1
1501—2000	Base	81.20%	0.18
	OPS	81.68%	0.19
3501—4000	Base	43.12%	0.55
	OPS	62.16%	0.36
5501—6000	Base	23.60%	0.64
	OPS	87.76%	0.13

(a) Accuracy and calibration error (CE) values of the base model and OPS for indicated values of t.

Figure 3: The adaptive behavior of Online Platt scaling (OPS) for the covariate drift dataset described in Section 1.2. The title of each panel indicates the time-window that panel corresponds to. The histogram of X_t values in the corresponding time window is plotted with maximum height normalized to 1. Also plotted is the true curve for $\Pr(Y = 1 \mid X = x)$ and two predictive curves: a base model trained on $t = 1$ to $t = 1000$, and OPS-calibrated models with parameter values fixed at the start of the time window. The base model is accurate for the training data which is mostly in $[-5, 10]$, but becomes inaccurate and miscalibrated with the covariate-shifted values for larger t (bottom two subplots). OPS adapts well, agreeing with the base model in the top-right subplot, but flipping the base model predictions in the bottom-right subplot.

1.2 Illustrative experiments with distribution drift

Covariate drift. We generated data as follows. For $t = 1, 2, \ldots, 6000$,

$$X_t \sim \mathcal{N}((t-1)/250, 4);$$
$$Y_t | X_t \sim \begin{cases} \text{Ber}(0.1) & \text{if mod}(\lfloor X_t/5 \rfloor, 2) = 0, \\ \text{Ber}(0.9) & \text{if mod}(\lfloor X_t/5 \rfloor, 2) = 1. \end{cases} \quad (3)$$

Thus the distribution of Y_t given X_t is a fixed periodic function, but the distribution of X_t drifts over time. The solid yellow line in Figure 3 plots $\Pr(Y = 1 \mid X = x)$ against x. We featurized x as a 48-dimensional vector with the components $\sin\left(\frac{x}{\text{freq}} + \text{translation}\right)$, where freq $\in \{1, 2, 3, 4, 5, 6\}$ and translation $\in \{0, \pi/4, \pi/2, \ldots 7\pi/4\}$.

A logistic regression base model f is trained over this 48-dimensional representation using the points $(X_t, Y_t)_{t=1}^{1000}$, randomly permuted and treated as a single batch of exchangeable points, which we will call *training points*. The points $(X_t, Y_t)_{t=1001}^{6000}$ form a supervised non-exchangeable test stream: we use this stream to evaluate f, recalibrate f using OPS, and evaluate the OPS-calibrated model.

Figure 3 displays f and the recalibrated OPS models at four ranges of t (one per plot). The training data has most x_t-values in the range $[-5, 10]$ as shown by the (height-normalized) histogram in the top-left plot. In this regime, f is visually accurate and calibrated—the dotted light blue line is close to the solid yellow truth. We now make some observations at three test-time regimes of t:

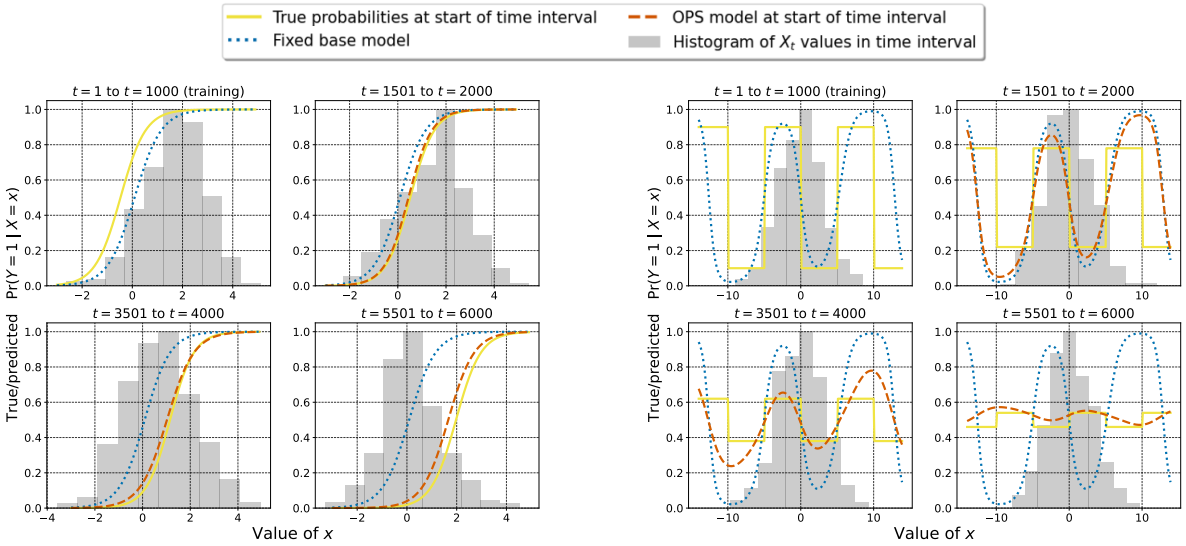
(a) $t = 1501$ to $t = 2000$ (the histogram shows the distribution of $(x_t)_{t=1501}^{2000}$). For these values of t, the test data is only slightly shifted from the training data, and f continues to perform well. The OPS model recognizes the good performance of f and does not modify it much.

(b) $t = 3500$ to $t = 4000$. Here, f is "out-of-phase" with the true distribution, and Platt scaling is insufficient to improve f by a lot. OPS recognizes this, and it offers slightly better calibration and accuracy by making less confident predictions between 0.2 and 0.4.

(c) $t = 5500$ to $t = 6000$. In this regime, f makes predictions opposing reality. Here, the OPS model flips the prediction, achieving high accuracy and calibration.

These observations are quantitatively supported by the accuracy and ℓ_1-calibration error (CE) values reported by the table in Figure 3a. Accuracy and CE values are estimated using the known true distribution of $Y_t \mid X_t$ and the observed X_t values, making them unbiased and avoiding some well-known issues with CE estimation. More details are provided in Appendix A.2.

Label drift. For $t = 1, 2, \ldots, 6000$, data is generated as:



(a) OPS with label drift.

(b) OPS with regression-function drift.

Figure 4: The adaptive behavior of OPS for the simulated label shift and regression-function drift datasets described in Section 1.2. For more details on the contents of the figure, please refer to Figure 3. The improvement in calibration and accuracy of OPS over the base model is visually apparent, but for completeness, {Acc, CE} values are reported in the Appendix as part of Figures 10 and 11.

$$\begin{aligned} Y_t &\sim \text{Bernoulli}(0.95(1-\alpha_t) + 0.05\alpha_t), \\ &\text{where } \alpha_t = (t-1)/6000); \\ X_t | Y_t &\sim \mathbb{1}\{Y_t = 0\}\mathcal{N}(0,1) + \mathbb{1}\{Y_t = 1\}\mathcal{N}(2,1). \end{aligned} \tag{4}$$

Thus, $X_t \mid Y_t$ is fixed while the label distribution drifts. We follow the same training and test splits described in the covariate drift experiment, but without sinusoidal featurization of X_t; the base logistic regression model is trained directly on the scalar X_t's. The gap between f and the true model increases over time but OPS adapts well (Figure 4a).

Regression-function drift. For $t = 1, 2, \ldots, 6000$, the data is generated as follows: $\alpha_t = (t-1)/5000$,

$$X_t \sim \mathcal{N}(0, 10) \text{ and } Y_t | X_t \sim \text{Bernoulli}(p_t), \text{ where} \tag{5}$$
$$p_t = \begin{cases} 0.1(1-\alpha_t) + 0.5\alpha_t & \text{if mod}(\lfloor X_t/5 \rfloor, 2) = 0, \\ 0.9(1-\alpha_t) + 0.5\alpha_t & \text{if mod}(\lfloor X_t/5 \rfloor, 2) = 1. \end{cases}$$

Thus the distribution of X_t is fixed, but the regression function $\Pr(Y_t = 1 \mid X_t)$ drifts over time. We follow the same training and test splits described in the covariate drift experiment, as well as the 48-dimensional featurization and logistic regression modeling. The performance of the base model worsens over time, while OPS adapts (Figure 4b).

2 Online Platt scaling (OPS)

In a batch post-hoc setting, the Platt scaling parameters are set to those that minimize log-loss over the calibration data. If we view the first t instances in our stream as the calibration data, the fixed-batch Platt scaling parameters are,

$$(\widehat{a}_t, \widehat{b}_t) = \operatorname*{arg\,min}_{(a,b)\in\mathbb{R}^2} \sum_{s=1}^{t} l(m^{a,b}(f(\mathbf{x}_s)), y_s), \tag{6}$$

where $l(p, y) = -y \log p - (1-y)\log(1-p)$ and $m^{a,b}$ is defined in (2). Observe that this is exactly logistic regression over the dataset $(\text{logit}(f(\mathbf{x}_s)), y_s)_{s=1}^t$.

The thesis of OPS is that as more data is observed over time, we should use it to update the Platt scaling parameters. Define $p_t^{\text{OPS}} := m^{a_t,b_t}(f(\mathbf{x}_t))$, where (a_t, b_t) depends on $\{(f(\mathbf{x}_1), y_1), \ldots, (f(\mathbf{x}_{t-1}), y_{t-1})\}$.[1] One way to compare methods in this online setting is to consider *regret* R_T with respect to a reference ℓ_2-ball of radius B, $\mathcal{B} := \{(a, b) \in \mathbb{R}^2 : a^2 + b^2 \leqslant B^2\}$:

$$R_T = \sum_{t=1}^{T} l(p_t^{\text{OPS}}, y_t) - \min_{(a,b)\in\mathcal{B}} \sum_{t=1}^{T} l(m^{a,b}(f(\mathbf{x}_t)), y_t). \tag{7}$$

R_T is the difference between the total loss incurred when playing (a_t, b_t) at times $t \leqslant T$ and the total loss incurred when playing the single optimal $(a, b) \in \mathcal{B}$ for all $t \leqslant T$. Typically, we are interested in algorithms that have low R_T irrespective of how $(\mathbf{x}_t, y_t)$ is generated.

2.1 Logarithmic worst-case regret bound for OPS

OPS regret minimization is exactly online logistic regression (OLR) regret minimization over "pseudo-features" $\text{logit}(f(\mathbf{x}_t))$. Thus our OPS problem is immediately solved

[1] A variant of this setup allows (a_t, b_t) to depend on $f(\mathbf{x}_t)$ (Foster et al., 2018).

Algorithm	Regret	Running time
Online Gradient Descent (OGD) (Zinkevich, 2003)	$B\sqrt{T}$	T
Online Newton Step (ONS) (Hazan et al., 2007)	$e^B \log T$	T
AIOLI (Jézéquel et al., 2020)	$B \log(BT)$	$T \log T$
Aggregating Algorithm (AA) (Vovk, 1990; Foster et al., 2018)	$\log(BT)$	$B^{18}T^{24}$

Table 1: Asymptotic regret and running times of online logistic regression (OLR) algorithms for OPS as functions of the radius of reference class B and time-horizon T. For general OLR, regret and running times also depend on the dimension of $\mathcal{X}$. However, OPS effectively reduces the dimensionality of $\mathcal{X}$ to 2, so that a second-order method like ONS runs almost as fast as a first-order method like OGD. Also note that $B = \sqrt{a^2 + b^2}$ is small if the base model f is not highly miscalibrated. ONS with fixed hyperparameters was chosen for all OPS experiments; see Section 2.2 for implementation details.

using OLR methods. A number of OLR methods have been proposed, and we consider their regret guarantees and running times for the OPS problem. These bounds typically depend on T and two problem-dependent parameters: the dimension (say d) and B, the radius of $\mathcal{B}$.

1. In our case, $d = 2$ since there is one feature $\text{logit}(f(\mathbf{x}))$ and a bias term. Thus d is a constant.
2. B could technically be large, but in practice, if f is not highly miscalibrated, we expect small values of a and b which would in turn lead to small B. This was true in all our experiments.

Regret bounds and running times for candidate OPS methods are presented in Table 1, which is an adaptation of Table 1 of Jézéquel et al. (2020) with all $\text{poly}(d)$ terms removed. Based on this table, we identify AIOLI and Online Newton Step (ONS) as the best candidates for implementing OPS, since they both have $O(\log T)$ regret and $\widetilde{O}(T)$ running time. In the following theorem, we collect explicit regret guarantees for OPS based on ONS and AIOLI. Since the log-loss can be unbounded if the predicted probability equals 0 or 1, we require some restriction on $f(\mathbf{x}_t)$.

Theorem 2.1. *Suppose* $\forall t, f(\mathbf{x}_t) \in [0.01, 0.99]$, $B \geqslant 1$, *and* $T \geqslant 10$. *Then, for any sequence* $(\mathbf{x}_t, y_t)_{t=1}^T$, *OPS based on ONS satisfies*

$$R_T(\textit{ONS}) \leqslant 2(e^B + 10B)\log T + 1, \tag{8}$$

and OPS based on AIOLI satisfies

$$R_T(\textit{AIOLI}) \leqslant 22B\log(BT). \tag{9}$$

The proof is in Appendix E. Since log-loss is a proper scoring rule (Gneiting and Raftery, 2007), minimizing it has implications for calibration (Bröcker, 2009). However, no "absolute" calibration bounds can be shown for OPS, as discussed shortly in Section 2.4.

2.2 Hyperparameter-free ONS implementation

In our experiments, we found ONS to be significantly faster than AIOLI while also giving better calibration. Further, ONS worked without any hyperparameter tuning after an initial investigation was done to select a single set of hyperparameters. Thus we used ONS for experiments based on a *verbatim implementation* of Algorithm 12 in Hazan (2016), with $\gamma = 0.1$, $\rho = 100$, and $\mathcal{K} = \{(a, b) : \|(a, b)\|_2 \leqslant 100\}$. Algorithm 1 in the Appendix contains pseudocode for our final OPS implementation.

2.3 Follow-The-Leader as a baseline for OPS

The Follow-The-Leader (FTL) algorithm sets $(a_t, b_t) = (\widehat{a}_{t-1}, \widehat{b}_{t-1})$ (defined in (6)) for $t \geqslant 1$. This corresponds to solving a logistic regression optimization problem at every time step, making the overall complexity of FTL $\Omega(T^2)$. Further, FTL has $\Omega(T)$ worst-case regret. Since full FTL is intractably slow to implement even for an experimental comparison, we propose to use a computationally cheaper variant, called Windowed Platt Scaling (WPS). In WPS the optimal parameters given all current data, $(\widehat{a}_t, \widehat{b}_t)$, are computed and updated every $O(100)$ steps instead of at every time step. We call this a *window* and the exact size of the window can be data-dependent. The optimal parameters computed at the start of the window are used to make predictions until the end of that window, then they are updated for the next window. This heuristic version of FTL performs well in practice (Section 4).

2.4 Limitations of regret analysis

Regret bounds are relative to the best in class, so Theorem 2.1 implies that OPS will do no worse than the best Platt scaling model in hindsight. However, even for i.i.d. data, the best Platt scaling model is itself miscalibrated on some distributions (Gupta et al., 2020, Theorem 3). This latter result shows that some form of binning must be deployed to be calibrated for arbitrarily distributed i.i.d. data. Further, if the data is adversarial, any deterministic predictor can be rendered highly miscalibrated (Oakes, 1985; Dawid, 1985); a simple strategy is to set $y_t = \mathbb{1}\{p_t \leqslant 0.5\}$. In a surprising seminal result, Foster and Vohra (1998) showed that adversarial calibration is possible by randomizing/hedging between different bins. The following section shows how one can perform such binning and hedging on top of OPS,

based on a technique called calibeating.

3 Calibeating the OPS forecaster

Calibeating (Foster and Hart, 2023) is a technique to improve or "beat" an expert forecaster. The idea is to first use the expert's forecasts to allocate data to representative *bins*. Then, the bins are treated *nominally*: they are just names or tags for "groups of data-points that the expert suggests are similar". The final forecasts in the bins are computed using only the outcome (y_t) values of the points in the bin (seen so far), with no more dependence on the expert's original forecast. The intuition is that forecasting inside each bin can be done in a theoretically valid sense, irrespective of the theoretical properties of the expert.

We will use the following "ϵ-bins" to perform calibeating:

$$B_1 = [0, \epsilon), B_2 = [\epsilon, 2\epsilon), \ldots, B_m = [1-\epsilon, 1]. \quad (10)$$

Here $\epsilon > 0$ is the width of the bins, and for simplicity we assume that $m = 1/\epsilon$ is an integer. For instance, one could set $\epsilon = 0.1$ or the number of bins $m = 10$, as we do in the experiments in Section 4. Two types of calibeating—tracking and hedging—are described in the following subsections. We suggest recalling our illustration of calibeating in the introduction (Figure 1c).

3.1 Calibeating via tracking past outcomes in bins

Say at some t, the expert forecasts $p_t \in [0.7, 0.8)$. We look at the instances $s < t$ when $p_s \in [0.7, 0.8)$ and compute

$$\bar{y}^b_{t-1} = \text{Average}\{y_s : s < t, p_s \in [0.7, 0.8)\}.$$

Suppose we find that $\bar{y}^b_{t-1} = 0.85$. That is, when the expert forecasted bin $[0.7, 0.8)$ in the past, the average outcome was 0.85. A natural idea now is to forecast 0.85 instead of 0.75. We call this process "Tracking", and it is the form of calibeating discussed in Section 4 of Foster and Hart (2023). In our case, we treat OPS as the expert and call the tracking version of OPS as TOPS. If $p^{\text{OPS}}_t \in B_b$, then

$$p^{\text{TOPS}}_t := \text{Average}\{y_s : s < t, p^{\text{OPS}}_s \in B_b\}. \quad (11)$$

The average is defined as the mid-point of B_b if the set above is empty.

Foster and Hart (2023) showed that the Brier-score of the TOPS forecasts p^{TOPS}_t, defined as $\frac{1}{T}\sum_{t=1}^{T}(y_t - p^{\text{TOPS}}_t)^2$, is better than the corresponding Brier-score of the OPS forecasts p^{OPS}_t, by roughly the squared calibration error of p^{OPS}_t (minus a $\log T$ term). In the forthcoming Theorem 3.1, we derive a result for a different object that is often of interest in post-hoc calibration, called sharpness.

3.2 Segue: defining sharpness of forecasters

Recall the ϵ-bins introduced earlier (10). Define $N_b = |\{t \leqslant T : p_t \in B_b\}|$ and $\widehat{y}_b = \frac{1}{N_b}\sum_{t \leqslant T, p_t \in B_b} y_t$ if $N_b > 0$, else $\widehat{y}_b = 0$. Sharpness is defined as,

$$\text{SHP}(p_{1:T}) := \frac{1}{T}\sum_{b=1}^{m} N_b \cdot \widehat{y}_b^2.^{2} \quad (12)$$

If the forecaster is perfectly knowledgeable and forecasts $p_t = y_t$, its SHP equals $\sum_{t=1}^{T} y_t/T =: \bar{y}_T$. On the other hand, if the forecaster puts all points into a single bin b, its SHP equals $(\sum_{t=1}^{T} y_t/T)^2 = \bar{y}_T^2$. The former forecaster is precise or *sharp*, while the latter is not, and SHP captures this—it can be shown that $\bar{y}_T^2 \leqslant \text{SHP}(p_{1:T}) \leqslant \bar{y}_T$. We point the reader to Bröcker (2009) for further background. One of the goals of effective forecasting is to ensure high sharpness (Gneiting et al., 2007). OPS achieves this goal by relying on the log-loss, a proper scoring rule. The following theorem shows that TOPS suffers a small loss in SHP compared to OPS.

Theorem 3.1. *The sharpness of TOPS forecasts satisfies*

$$\textit{SHP}(p^{\textit{TOPS}}_{1:T}) \geqslant \textit{SHP}(p^{\textit{OPS}}_{1:T}) - \epsilon - \frac{\epsilon^2}{4} - \frac{\log T + 1}{\epsilon T}. \quad (13)$$

The proof (in Appendix E) uses Theorem 3 of Foster and Hart (2023) and relationships between sharpness, Brier-score, and a quantity called refinement. If T is fixed and known, setting $\epsilon \approx \sqrt{\log T/T}$ (including constant factors), or equivalently, the number of bins $B \approx \sqrt{T/\log T}$ gives a rate of $\widetilde{O}(\sqrt{1/T})$ for the SHP difference term. While we do not show a calibration guarantee, TOPS had the best calibration performance in most experiments (Section 4)

3.3 Calibeating via hedging or randomized prediction

All forecasters introduced so far—the base model f, OPS, and TOPS—make forecasts p_t that are deterministic given the past data until time $t-1$. If the y_t sequence is being generated by an adversary that acts after seeing p_t, then the adversary can ensure that each of these forecasters is miscalibrated by setting $y_t = \mathbb{1}\{p_t \leqslant 0.5\}$.

Suppose instead that the forecaster is allowed to hedge—randomize and draw the forecast from a distribution instead of fixing it to a single value—and the adversary only has access to the distribution and not the actual p_t. Then there exist hedging strategies that allow the forecaster to be arbitrarily well-calibrated (Foster and Vohra, 1998). In fact, Foster (1999, henceforth F99) showed that this can be done while hedging between two arbitrarily close points in $[0, 1]$.

In practice, outcomes are not adversarial, and covariates are available. A hedging algorithm that does not use covariates cannot be expected to give informative predictions. We verify this intuition through an experiment in Appendix on historical rain data D—F99's hedging algorithm simply predicts the average y_t value in the long run.

[2] The original definition of sharpness (Murphy, 1973) was (essentially): $-T^{-1}\sum_{b=1}^{m} N_b \widehat{y}_b(1-\widehat{y}_b)$, which equals $\text{SHP}(p_{1:T}) - \bar{y}_T$. We add the forecast-independent term $\bar{y}_T$ on both sides and define the (now non-negative) quantity as SHP.

A best-of-both-worlds result can be achieved by using the expert forecaster to bin data using $\mathbf{x}_t$ values, just like we did in Section 3.1. Then, inside every bin, a separate hedging algorithm is instantiated. For the OPS predictor, this leads to HOPS (OPS + hedging). Specifically, in our experiments and the upcoming calibration error guarantee, we used F99:

$$p_t^{\text{HOPS}} := \text{F99}(y_s : s < t, p_s \in B_b). \tag{14}$$

A standalone description of F99 is included in Appendix C. F99 hedges between consecutive mid-points of the ϵ-bins defined earlier (10). The only hyperparameter for F99 is ϵ. In the experiments in the main paper, we set $\epsilon = 0.1$. To be clear, p_t is binned on the ϵ-bins, and the hedging inside each bin is again over the ϵ-bins.

The upcoming theorem shows a SHP lower bound on HOPS. In addition, we show an assumption-free upper bound on the (ℓ_1-)calibration error, defined as

$$\text{CE}(p_{1:T}) := \frac{1}{T}\sum_{b=1}^{m} N_b \cdot |\widehat{p}_b - \widehat{y}_b|\,, \tag{15}$$

where $N_b, \widehat{y}_b$ were defined in Section 3.2, and $\widehat{p}_b = \frac{1}{N_b}\sum_{t\leqslant T, p_t\in B_b} p_t$, if $N_b > 0$, else $\widehat{p}_b = \text{mid-point}(B_b)$. Achieving small CE is one formalization of (1). The following result is conditional on the $y_{1:T}$, $p^{\text{OPS}}_{1:T}$ sequences. The expectation is over the randomization in F99.

Theorem 3.2. *For adversarially generated data, the expected sharpness of HOPS forecasts using the forecast hedging algorithm of Foster (1999) is lower bounded as*

$$\mathbb{E}\left[\mathit{SHP}(p^{\mathit{HOPS}}_{1:T})\right] \geqslant \mathit{SHP}(p^{\mathit{OPS}}_{1:T}) - \left(\epsilon + \frac{\log T + 1}{\epsilon^2 T}\right), \tag{16}$$

and the expected calibration error of HOPS satisfies,

$$\mathbb{E}\left[\mathit{CE}(p^{\mathit{HOPS}}_{1:T})\right] \leqslant \epsilon/2 + 2\sqrt{1/\epsilon^2 T}. \tag{17}$$

The proof in Appendix E is based on Theorem 5 of Foster and Hart (2023) and a CE bound for F99 based on Blackwell approachability (Blackwell, 1956). With $\epsilon = \widetilde{\Theta}(T^{-1/3})$, the difference term in the SHP bound is $\widetilde{O}(T^{-1/3})$ and with $\epsilon = \widetilde{\Theta}(T^{-1/4})$, the CE bound is $\widetilde{O}(T^{-1/4})$. Compare this to the usual CE rate without calibeating of $O(T^{-1/3})$ (Foster and Vohra, 1998). High-probability versions of (17) can be derived using probabilistic Blackwell approachability lemmas, such as those in Perchet (2014).

4 Experiments

We perform experiments with synthetic and real-data, in i.i.d. and distribution drift setting. Code to reproduce the experiments can be found at `https://github.com/aigen/df-posthoc-calibration` (see Appendix A.4 for more details). All baseline and proposed methods are described in Collection 1 on the following page. In each experiment, the **base model** f was a random forest (`sklearn`'s implementation). All default parameters were used, except `n_estimators` was set to 1000. No hyperparameter tuning on individual datasets was performed for any of the recalibration methods.

Collection 1. Proposed and baseline methods for online post-hoc calibration. Final forecasts are identified in blue.

Input: $f : \mathcal{X} \to [0,1]$, any pre-learnt model
Input: $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_T, y_T) \in \mathcal{X} \times \{0,1\}$
Input: calibration-set-size $T_{\text{cal}} < T$, window-size W
Fixed Platt scaling: $(a^{\text{FPS}}, b^{\text{FPS}}) \leftarrow (\widehat{a}_{T_{\text{cal}}}, \widehat{b}_{T_{\text{cal}}})$ (eq. 6)
Windowed Platt scaling: $(a^{\text{WPS}}, b^{\text{WPS}}) \leftarrow (a^{\text{FPS}}, b^{\text{FPS}})$
Online Platt scaling: $(a_1^{\text{OPS}}, b_1^{\text{OPS}}) \leftarrow (1, 0)$
for $t = 2$ **to** T **do**
  $(a_t^{\text{OPS}}, b_t^{\text{OPS}}) \leftarrow \text{ONS}((\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_{t-1}, y_{t-1}))$
  (ONS is Algorithm 1 in the Appendix)
end for
for $t = T_{\text{cal}} + 1$ **to** T **do**
  $p_t^{\text{BM}} \leftarrow f(\mathbf{x}_t)$
  $p_t^{\text{FPS}} \leftarrow \text{sigmoid}(a^{\text{FPS}} \cdot \text{logit}(f(\mathbf{x}_t)) + b^{\text{FPS}})$
  $p_t^{\text{WPS}} \leftarrow \text{sigmoid}(a^{\text{WPS}} \cdot \text{logit}(f(\mathbf{x}_t)) + b^{\text{WPS}})$
  $p_t^{\text{OPS}} \leftarrow \text{sigmoid}(a_t^{\text{OPS}} \cdot \text{logit}(f(\mathbf{x}_t)) + b_t^{\text{OPS}})$
  p_t^{TOPS} is set using past (y_s, p_s^{OPS}) values as in (11)
  p_t^{HOPS} is set using past (y_s, p_s^{OPS}) values as in (14)
  If $\mod(t - T_{\text{cal}}, W) = 0$, $(a^{\text{WPS}}, b^{\text{WPS}}) \leftarrow (\widehat{a}_t, \widehat{b}_t)$
end for

Metrics. We measured the SHP and CE metrics defined in (12) and (15) respectively. Although estimating population versions of SHP and CE in statistical (i.i.d.) settings is fraught with several issues (Kumar et al. (2019); Roelofs et al. (2022) and several other works), our definitions target actual observed quantities which are directly interpretable without reference to population quantities.

Reading the plots. The plots we report show CE values at certain time-stamps starting from $T_{\text{cal}} + 2W$ and ending at T (see third line of Collection 1). T_{cal} and W are fixed separately for each dataset (Table 2 in Appendix). We also generated SHP plots, but these are not reported since the drop in SHP was always very small.

4.1 Experiments on real datasets

We worked with four public datasets in two settings. Links to the datasets are in Appendix A.1.

Distribution drift. We introduced synthetic drifts in the data based on covariate values, so this is an instance of covariate drift. For example, in the bank marketing dataset (leftmost plot in Figure 5), the problem is to predict which clients are likely to subscribe to a term deposit if they are targeted for marketing, using covariates like `age`, `education`, and `bank-balance`. We ordered the available 12000 rows roughly by `age` by adding a random num-

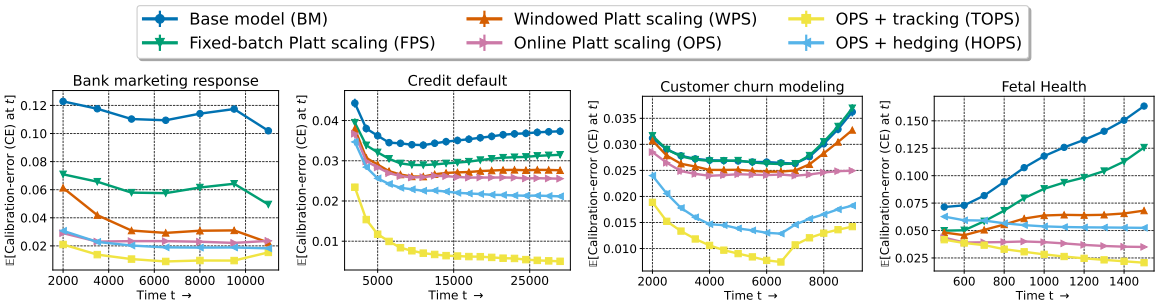


Figure 5: **Drifting data.** CE (calibration error) values over time of considered models on four datasets with synthetically induced drifts. The plots have invisible error bars since variation across the 100 runs was small. OPS consistently performs better than BM, FPS, and WPS, while TOPS is the best-performing among all methods across datasets and time. All methods had roughly the same SHP values at a given time-step, so the SHP plots are delayed to Appendix A (Figure 8).

ber uniformly from $\{-1, 0, 1\}$ to `age` and sorting all the data. Training is done on the first 1000 points, $T_{\text{cal}} = 1000$, and $W = 500$. Similar drifts are induced for the other datasets, and T_{cal}, W values are set depending on the total number of points; further details are in Appendix A.1.

All simulations were performed 100 times and the average CE and SHP values with $\pm$ std-deviation errorbars were evaluated at certain time-steps. Thus, our lines correspond to estimates of the expected values of CE and SHP, as indicated by the Y-axis labels. We find that across datasets, OPS has the least CE among non-calibeating methods, and both forms of calibeating typically improve OPS further (Figure 5). Specifically, TOPS performs the best by a margin compared to other methods. We also computed SHP values, which are reported in Appendix A (Figure 8). The drop in SHP is insignificant in each case (around 0.005).

IID data. This is the usual batch setting formed by shuffling all available data. Part of the data is used for training and the rest forms the test-stream. We used the same values of T_{cal} and W as those used in the data drift experiments (see Appendix A.1). In our experiments, we find that the gap in CE between BM, FPS, OPS, and WPS is smaller (Figure 6). However, TOPS performs the best in all scenarios, typically by a margin. Here too, the change in SHP was small, so those plots were delayed to Appendix A (Figure 9).

4.2 Synthetic experiments

In all experiments with real data, WPS performs almost as good as OPS. In this subsection, we consider some synthetic data drift experiments where OPS and TOPS continue performing well, but WPS performs much worse.

Covariate drift. Once for the entire process, we draw random orthonormal vectors $\mathbf{v}_1, \mathbf{v}_2 \in \mathbb{R}^{10}$ ($\|\mathbf{v}_1\|_2 = \|\mathbf{v}_2\|_2 = 1, \mathbf{v}_1^\intercal \mathbf{v}_2 = 0$), a random weight vector $\mathbf{w} \in \{-1, 1\}^{10+\binom{10}{2}}$ with each component set to 1 or -1 independently with probability 0.5, and set a drift parameter $\delta \geqslant 0$. The data is generated as follows:

$$\mathbf{u}_t = \mathbf{v}_1 \cos(\delta t) + \mathbf{v}_2 \sin(\delta t), X_t \sim \mathcal{N}(\mathbf{0}, \mathcal{I}_{10} + 10\mathbf{u}_t\mathbf{u}_t^\intercal),$$
$$Y_t | X_t \sim \text{Bernoulli}(\text{sigmoid}(\mathbf{w}^\intercal \widetilde{X}_t)), \text{ where}$$
$$\widetilde{X}_t = [\mathbf{x}_1, \ldots, \mathbf{x}_{10}, \mathbf{x}_1\mathbf{x}_2, \mathbf{x}_1\mathbf{x}_3, \ldots, \mathbf{x}_9\mathbf{x}_{10}] \in \mathbb{R}^{10+\binom{10}{2}}.$$

Thus the distribution of Y_t given X_t is fixed as a logistic model over the expanded representation $\widetilde{X}_t$ that includes all cross-terms (this is unknown to the forecaster who only sees X_t). The features X_t themselves are normally distributed with mean $\mathbf{0}$ and a time-varying covariance matrix. The principal component (PC) of the covariance matrix is a vector $\mathbf{u}_t$ that is rotating on the two-dimensional plane containing the orthonormal vectors $\mathbf{v}_1$ and $\mathbf{v}_2$. The first 1000 points are used as training data, the remaining $T = 5000$ form a test-stream, and $W = 500$. We report results in two settings: one is i.i.d., that is $\delta = 0$, and the other is where the $\mathbf{u}$ for the first and last point are at a 180° angle (Figure 7a).

Label drift. Given some $\delta > 0$, (X_t, Y_t) is generated as:

$$Y_t \sim \text{Bernoulli}(0.5 + \delta t),$$
$$X_t | Y_t \sim \mathbb{1}\{Y_t = 0\}\mathcal{N}(\mathbf{0}, \mathbb{R}^{10}) + \mathbb{1}\{Y_t = 1\}\mathcal{N}(\mathbf{e_1}, \mathbb{R}^{10}).$$

Thus $P(Y_1 = 1) = 0.5 + \delta$ and for the last test point, $P(Y_{6000} = 1) = 0.5 + 6000\delta$. This final value can be set to control the extent of label drift; we show results with no drift (i.e., $\delta = 0$, Figure 7b left) and δ set so that final bias $0.5 + 6000\delta = 0.9$ (Figure 7b right). The number of training points is 1000, $T = 5000$, and $W = 500$.

4.3 Changing ϵ, histogram binning, beta scaling

In Appendix A.3, we report versions of Figures 5, 6 with $\epsilon = 0.05, 0.2$ (instead of $\epsilon = 0.1$) with similar conclusions (Figures 12, 13). We also perform comparisons with a windowed version of the popular histogram binning method (Zadrozny and Elkan, 2001) and online versions of the beta scaling method, as discussed in the forthcoming Section 5.

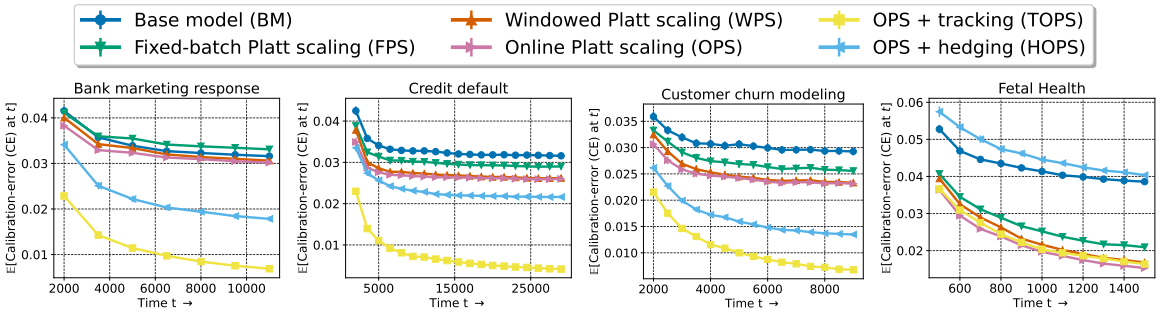
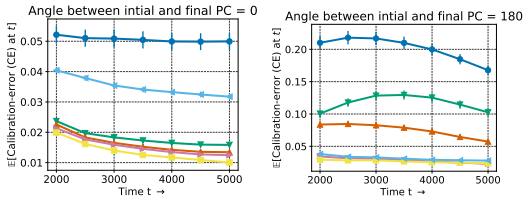
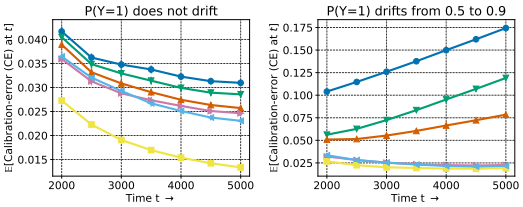


Figure 6: **IID data.** CE values over time of considered models with four randomly shuffled (ie, nearly i.i.d.) datasets. The plots have invisible error bars since variation across runs was small. TOPS achieves the smallest values of CE throughout.



(a) Left plot: i.i.d. data, right plot: covariate drift.



(b) Left plot: i.i.d. data, right plot: label drift.

Figure 7: Experiments with synthetic data. In all cases, TOPS has the lowest CE across time.

5 Online beta scaling with calibeating

A recalibration method closely related to Platt scaling is beta scaling (Kull et al., 2017). The beta scaling mapping m has three parameters $(a, b, c) \in \mathbb{R}^3$,

$$m_{\text{beta}}^{a,b,c}(f(\mathbf{x})) := \text{sigmoid}(a \cdot \log f(\mathbf{x}) + b \cdot \log(1 - f(\mathbf{x})) + c).$$

Observe that enforcing $b = -a$ recovers Platt scaling since $\text{logit}(z) = \log(z) - \log(1-z)$. The beta scaling parameters can be learnt following identical protocols as Platt scaling: (i) the traditional method of fixed batch post-hoc calibration akin to FPS, (ii) a natural benchmark of windowed updates akin to WPS, and (iii) regret minimization based method akin to OPS. This leads to the methods FBS, WBS, and OBS, replacing the "P" of Platt with the "B" of beta. Tracking + OBS (TOBS) and Hedging + OBS (HOBS) can be similarly derived. Further details on all beta scaling methods are in Appendix B, where we also report plots similar to Figures 5, 6 for beta scaling (Figure 14). In a comparison between histogram binning, beta scaling, Platt scaling, and their tracking versions, TOPS and TOBS are the best-performing methods across experiments (Figure 15).

6 Summary

We provided a way to bridge the gap between the online (typically covariate-agnostic) calibration literature, where data is assumed to be adversarial, and the (typically i.i.d.) post-hoc calibration literature, where the joint covariate-outcome distribution takes centerstage. The TOPS method we proposed has the lowest calibration error in all our experiments. The HOPS method provably controls miscalibration at any pre-defined level and could be a desirable choice in sensitive applications. The good performance of OPS+calibeating lends further empirical backing to the thesis that scaling+binning methods perform well in practice, as has also been noted in prior works (Zhang et al., 2020; Kumar et al., 2019).

We note a few directions for future work. First, online algorithms that control regret on the most recent data have been proposed (Hazan and Seshadhri, 2009; Zhang et al., 2018). These approaches could give further improvements on ONS, particularly for drifting data. Second, while this paper entirely discusses calibration for binary classification, all binary routines can be lifted to achieve multiclass notions such as top-label or class-wise calibration (Gupta and Ramdas, 2022b). Alternatively, multiclass versions of Platt scaling (Guo et al., 2017) such as temperature and vector scaling can also be targeted directly using online multiclass logistic regression (Jézéquel et al., 2021).

Acknowledgements

We thank Youngseog Chung and Dhruv Malik for fruitful discussions, and the ICML reviewers for valuable feedback. CG was supported by the Bloomberg Data Science Ph.D. Fellowship. For computation, we used allocation CIS220171 from the Advanced Cyberinfrastructure Coordination Ecosystem: Services & Support (ACCESS) program, supported by NSF grants 2138259, 2138286, 2138307, 2137603, and 2138296. Specifically, we used the Bridges-2 system (Towns et al., 2014), supported by NSF grant 1928147, at the Pittsburgh Supercomputing Center (PSC).

References

David Blackwell. An analog of the minimax theorem for vector payoffs. *Pacific Journal of Mathematics*, 6(1):1–8, 1956.

Jochen Bröcker. Reliability, sufficiency, and the decomposition of proper scores. *Quarterly Journal of the Royal Meteorological Society: A journal of the atmospheric sciences, applied meteorology and physical oceanography*, 135(643):1512–1519, 2009.

A Philip Dawid. The well-calibrated Bayesian. *Journal of the American Statistical Association*, 77(379):605–610, 1982.

A Philip Dawid. Comment: The impossibility of inductive inference. *Journal of the American Statistical Association*, 80(390):340–341, 1985.

Morris H DeGroot and Stephen E Fienberg. Assessing probability assessors: Calibration and refinement. Technical report, Carnegie Mellon University, 1981.

Dean P Foster. A proof of calibration via Blackwell's approachability theorem. *Games and Economic Behavior*, 29(1-2):73–78, 1999.

Dean P Foster and Sergiu Hart. Forecast hedging and calibration. *Journal of Political Economy*, 129(12):3447–3490, 2021.

Dean P Foster and Sergiu Hart. "Calibeating": beating forecasters at their own game. *Theoretical Economics (to appear)*, 2023.

Dean P Foster and Rakesh V Vohra. Asymptotic calibration. *Biometrika*, 85(2):379–390, 1998.

Dylan J Foster, Satyen Kale, Haipeng Luo, Mehryar Mohri, and Karthik Sridharan. Logistic regression: The importance of being improper. In *Conference On Learning Theory*, 2018.

Tilmann Gneiting and Adrian E Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American statistical Association*, 102(477):359–378, 2007.

Tilmann Gneiting, Fadoua Balabdaoui, and Adrian E Raftery. Probabilistic forecasts, calibration and sharpness. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 69(2):243–268, 2007.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International Conference on Machine Learning*, 2017.

Chirag Gupta and Aaditya Ramdas. Distribution-free calibration guarantees for histogram binning without sample splitting. In *International Conference on Machine Learning*, 2021.

Chirag Gupta and Aaditya Ramdas. Faster online calibration without randomization: interval forecasts and the power of two choices. In *Conference On Learning Theory*, 2022a.

Chirag Gupta and Aaditya Ramdas. Top-label calibration and multiclass-to-binary reductions. In *International Conference on Learning Representations*, 2022b.

Chirag Gupta, Aleksandr Podkopaev, and Aaditya Ramdas. Distribution-free binary classification: prediction sets, confidence intervals and calibration. In *Advances in Neural Information Processing Systems*, 2020.

Elad Hazan. Introduction to online convex optimization. *Foundations and Trends in Optimization*, 2(3-4):157–325, 2016.

Elad Hazan and Comandur Seshadhri. Efficient learning algorithms for changing environments. In *International Conference on Machine Learning*, 2009.

Elad Hazan, Amit Agarwal, and Satyen Kale. Logarithmic regret algorithms for online convex optimization. *Machine Learning*, 69(2):169–192, 2007.

Rémi Jézéquel, Pierre Gaillard, and Alessandro Rudi. Efficient improper learning for online logistic regression. In *Conference on Learning Theory*, 2020.

Rémi Jézéquel, Pierre Gaillard, and Alessandro Rudi. Mixability made efficient: Fast online multiclass logistic regression. In *Advances in Neural Information Processing Systems*, 2021.

Meelis Kull, Telmo M Silva Filho, and Peter Flach. Beyond sigmoids: How to obtain well-calibrated probabilities from binary classifiers with beta calibration. *Electronic Journal of Statistics*, 11(2):5052–5080, 2017.

Ananya Kumar, Percy S Liang, and Tengyu Ma. Verified uncertainty calibration. In *Advances in Neural Information Processing Systems*, 2019.

Allan H Murphy. A new vector partition of the probability score. *Journal of applied Meteorology*, 12(4):595–600, 1973.

Alexandru Niculescu-Mizil and Rich Caruana. Predicting good probabilities with supervised learning. In *International Conference on Machine Learning*, 2005.

David Oakes. Self-calibrating priors do not exist. *Journal of the American Statistical Association*, 80(390):339–339, 1985.

Vianney Perchet. Approachability, regret and calibration: Implications and equivalences. *Journal of Dynamics & Games*, 1(2):181, 2014.

John Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, 10(3):61–74, 1999.

Rebecca Roelofs, Nicholas Cain, Jonathon Shlens, and Michael C Mozer. Mitigating bias in calibration error estimation. In *International Conference on Artificial Intelligence and Statistics*, 2022.

J. Towns, T. Cockerill, M. Dahan, I. Foster, K. Gaither, A. Grimshaw, V. Hazlewood, S. Lathrop, D. Lifka, G. D. Peterson, R. Roskies, J. Scott, and N. Wilkins-Diehr. XSEDE: Accelerating Scientific Discovery. *Computing in Science & Engineering*, 16(5):62–74, 2014.

Volodimir G Vovk. Aggregating strategies. In *Conference on Computational Learning Theory*, 1990.

David Widmann, Fredrik Lindsten, and Dave Zachariah. Calibration tests in multi-class classification: A unifying framework. In *Advances in Neural Information Processing Systems*, 2019.

Bianca Zadrozny and Charles Elkan. Obtaining calibrated probability estimates from decision trees and naive Bayesian classifiers. In *International Conference on Machine Learning*, 2001.

Bianca Zadrozny and Charles Elkan. Transforming classifier scores into accurate multiclass probability estimates. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2002.

Jize Zhang, Bhavya Kailkhura, and T Yong-Jin Han. Mix-n-match: Ensemble and compositional methods for uncertainty calibration in deep learning. In *International Conference on Machine Learning*, 2020.

Lijun Zhang, Tianbao Yang, and Zhi-Hua Zhou. Dynamic regret of strongly adaptive methods. In *International Conference on Machine Learning*, 2018.

Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *International Conference on Machine Learning*, 2003.

Name	T_{train}	T_{cal}	W	Sort-by	Link to dataset
Bank marketing	1000	1000	500	Age	https://www.kaggle.com/datasets/kukuroo3/bank-marketing-response-predict
Credit default	1000	1000	500	Sex	https://www.kaggle.com/datasets/uciml/default-of-credit-card-clients-dataset
Customer churn	1000	1000	500	Location	https://www.kaggle.com/datasets/shrutimechlearn/churn-modelling
Fetal health	626	300	100	Acceleration	https://www.kaggle.com/datasets/andrewmvd/fetal-health-classification

Table 2: Metadata for datasets used in Section 4.1. The sort-by column indicates which covariate was used to order data points. All datasets are under the Creative Commons CC0 license.

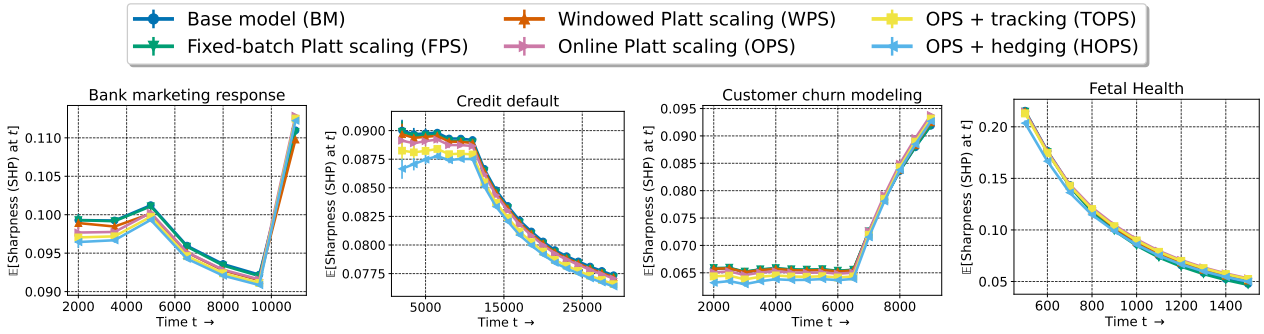


Figure 8: **Sharpness results with drifting data.** SHP values over time of considered models on four datasets with synthetically induced drifts (Section 4.1). The plots have invisible error bars since variation across the 100 runs was small. The drop in expected sharpness is below 0.005 at all times except on the Fetal Health Dataset.

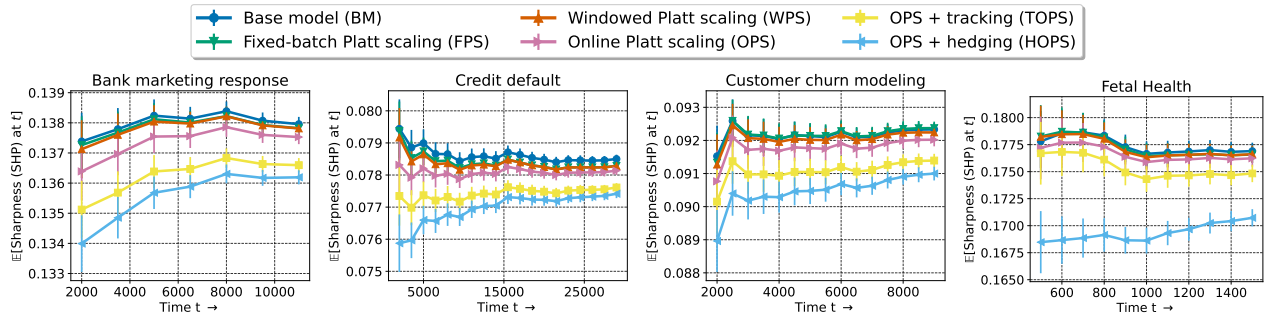


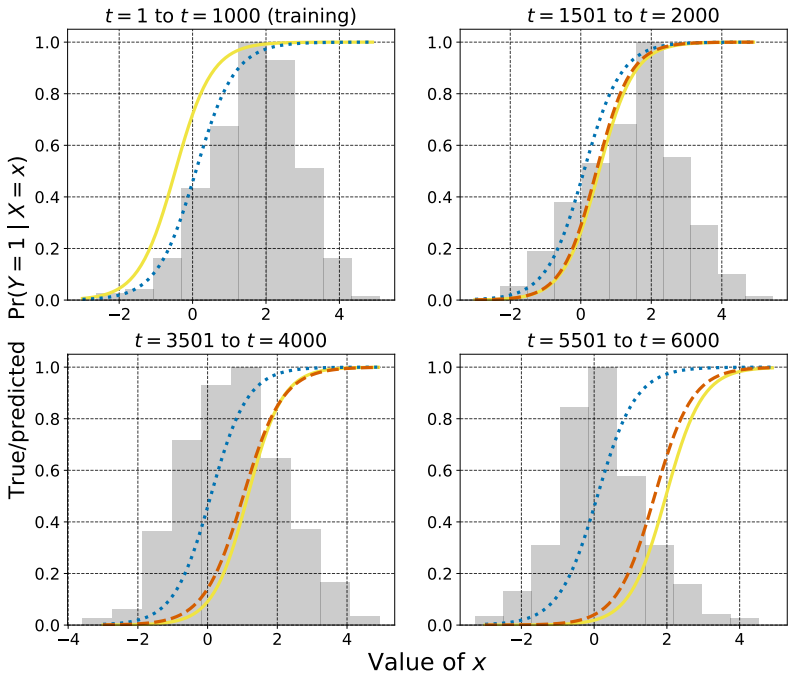
Figure 9: **Sharpness results with i.i.d. data.** SHP values over time of considered models on four shuffled (ie, nearly i.i.d.) datasets (Section 4.1). The drop in expected sharpness is less than 0.005 in all cases except for the HOPS forecaster on the Fetal Health dataset, where it is 0.01.

A Experimental details and additional results

Some implementation details, metadata, information on metrics, and additional results and figures are collected here.

A.1 Metadata for datasets used in Section 4.1

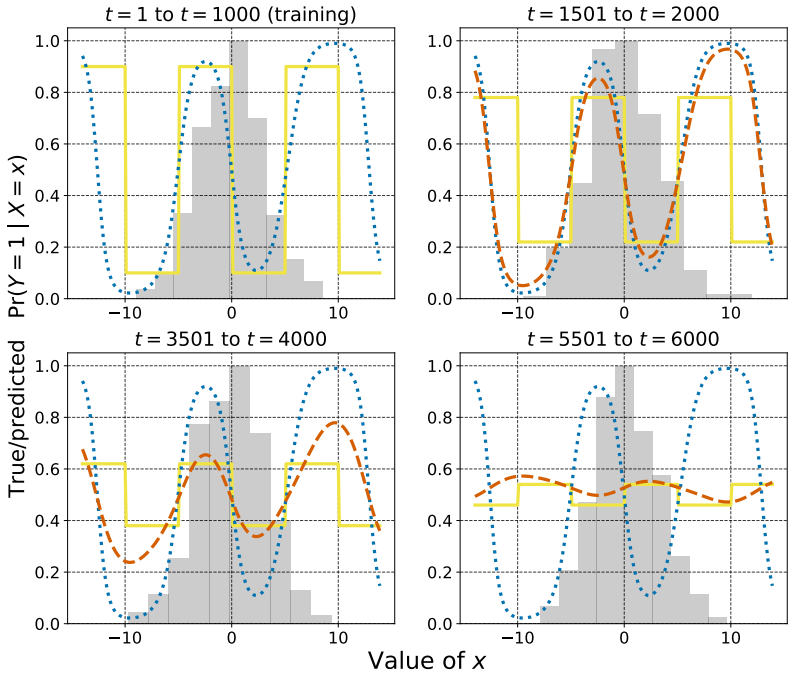
Table 2 contains metadata for the datasets we used in Section 4.1. T_{train} refers to the number of training examples. The "sort-by" column indicates which covariate was used to order data points. In each case some



t	Model	Acc ↑	CE ↓
0—1000	Base	90.72%	0.025
1500—2000	Base	84.29%	0.088
	OPS	86.08%	0.029
3500—4000	Base	68.80%	0.26
	OPS	83.07%	0.053
5500—6000	Base	59.38%	0.42
	OPS	92.82%	0.049

(a) Table reporting accuracy and CE values of the base model and OPS for indicated values of t.

Figure 10: The adaptive behavior of OPS for the simulated label drift scenario described in Section 1.2.



t	Model	Acc ↑	CE ↓
0—1000	Base	84.14%	0.1
1500—2000	Base	74.64%	0.12
	OPS	75.40%	0.097
3500—4000	Base	59.51%	0.2
	OPS	59.60%	0.067
5500—6000	Base	44.39%	0.34
	OPS	51.43%	0.049

(a) Table reporting accuracy and CE values of the base model and OPS for indicated values of t.

Figure 11: The adaptive behavior of OPS for the simulated regression-function drift scenario described in Section 1.2.

noise was added to the covariate in order to create variation for the experiments. The exact form of drift can be found in the python file `sec_4_experiments_core.py` in the repository https://github.com/AIgen/df-posthoc-calibration/tree/main/Online%20Platt%20Scaling%20with%20Calibeating.

A.2 Additional plots and details for label drift and regression-function drift experiments from Section 1

Figures 3, 10, and 11 report accuracy (Acc) and calibration error (CE) values for the base model and the OPS model in the three dataset drift settings we considered. The Acc values are straightforward averages and can be computed without issues. However, estimation of CE on real datasets is tricky and requires sophisticated techniques such as adaptive binning, debiasing, heuristics for selecting numbers of bins, or kernel estimators (Kumar et al., 2019; Roelofs et al., 2022; Widmann et al., 2019). The issue typically boils down to the fact that $\Pr(Y = 1 \mid X = x)$ cannot be estimated for every $x \in \mathcal{X}$ without making smoothness assumptions or performing some kind of binning. However, in the synthetic experiments of Section 1, $\Pr(Y = 1 \mid X)$ is known exactly, so such techniques are not required. For some subset of forecasts $p_s, p_2, \ldots, p_t$, we compute

$$\text{CE} = \frac{1}{t-s+1} \sum_{i=s}^{t} \left| p_i - \Pr(Y_i = 1 \mid X_i = \mathbf{x}_i) \right|,$$

on the instantiated values of $X_s, X_{s+1}, \ldots, X_t$. Thus, what we report is the true CE given covariate values.

A.3 Additional results with windowed histogram binning and changing bin width

Comparison to histogram binning (HB). HB is a recalibration method that has been shown to have excellent empirical performance as well as theoretical guarantees (Zadrozny and Elkan, 2001; Gupta and Ramdas, 2021). There are no online versions of HB that we are aware of, so we use the same windowed approach as windowed Platt and beta scaling for benchmarking (see Section 2.3 and the second bullet in Section B). This leads to windowed histogram binning (WHB), the fixed-batch HB recalibrator that is updated every $O(100)$ time-steps. We compare WHB to OPS and OBS (see Section 5). Since tracking improves both OPS and OBS, we also consider tracking WHB. Results are presented in Figure 15.

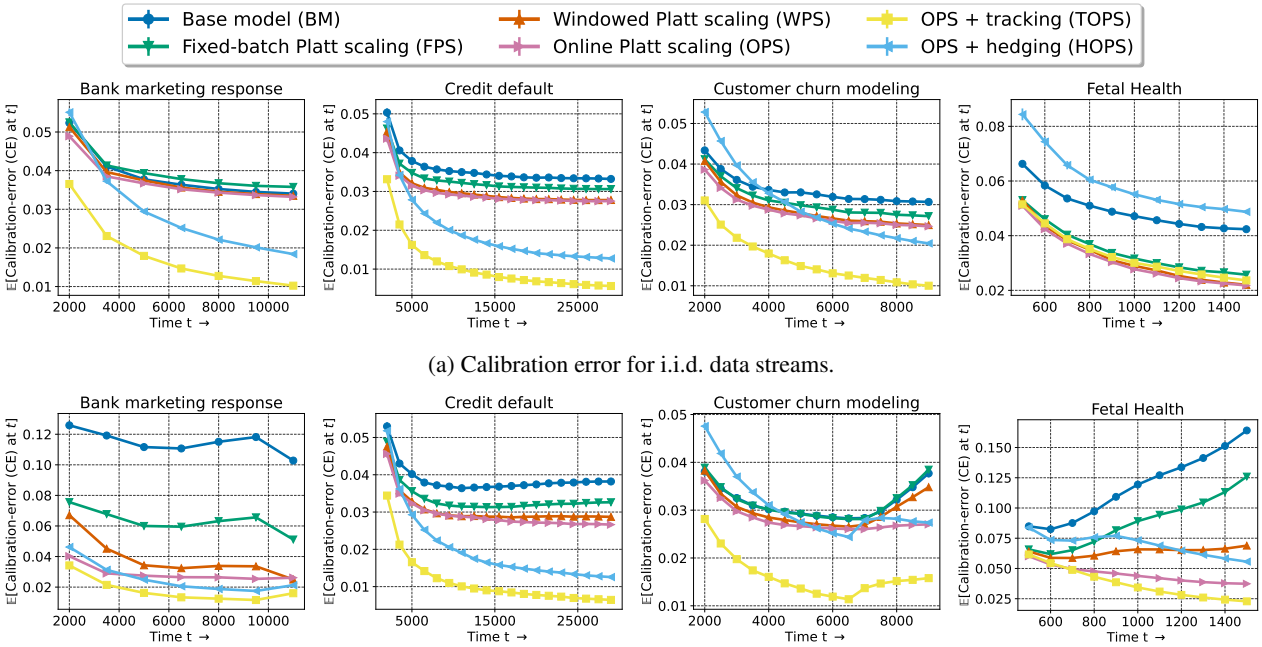
We find that WHB often performs better than OPS and OBS in the i.i.d. case, and results are mixed in the drifting case. However, since WHB is a binning method, it inherently produces something akin to a running average, and so tracking does not improve it further. The best methods (TOPS, TOBS) are the ones that combine one of our proposed parametric online calibrators (OPS, OBS) with tracking.

Changing the bin width ϵ. In the main paper, we used $\epsilon = 0.1$ and defined corresponding bins as in (10). This binning reflects in three ways on the experiments we performed. First, ϵ-binning is used to divide forecasts into representative bins before calibeating (equations (11), (14)). Second, ϵ-binning is used to define the sharpness and calibration error metrics. Third, the hedging procedure F99 requires specifying a binning scheme, and we used the same ϵ-bins.

Here, we show that the empirical results reported in the main paper are not sensitive to the chosen representative value of $\epsilon = 0.1$. We run the same experiment used to produce Figures 5 and 6 but with $\epsilon = 0.05$ (Figure 12) and $\epsilon = 0.2$ (Figure 13). The qualitative results remain identical, with TOPS still the best performer and hardly affected by the changing epsilon. In fact, the plots for all methods except HOPS are indistinguishable from their $\epsilon = 0.1$ counterparts at first glance. HOPS is slightly sensitive to ϵ: the performance improves slightly with $\epsilon = 0.05$, and worsens slightly with $\epsilon = 0.2$.

A.4 Reproducibility

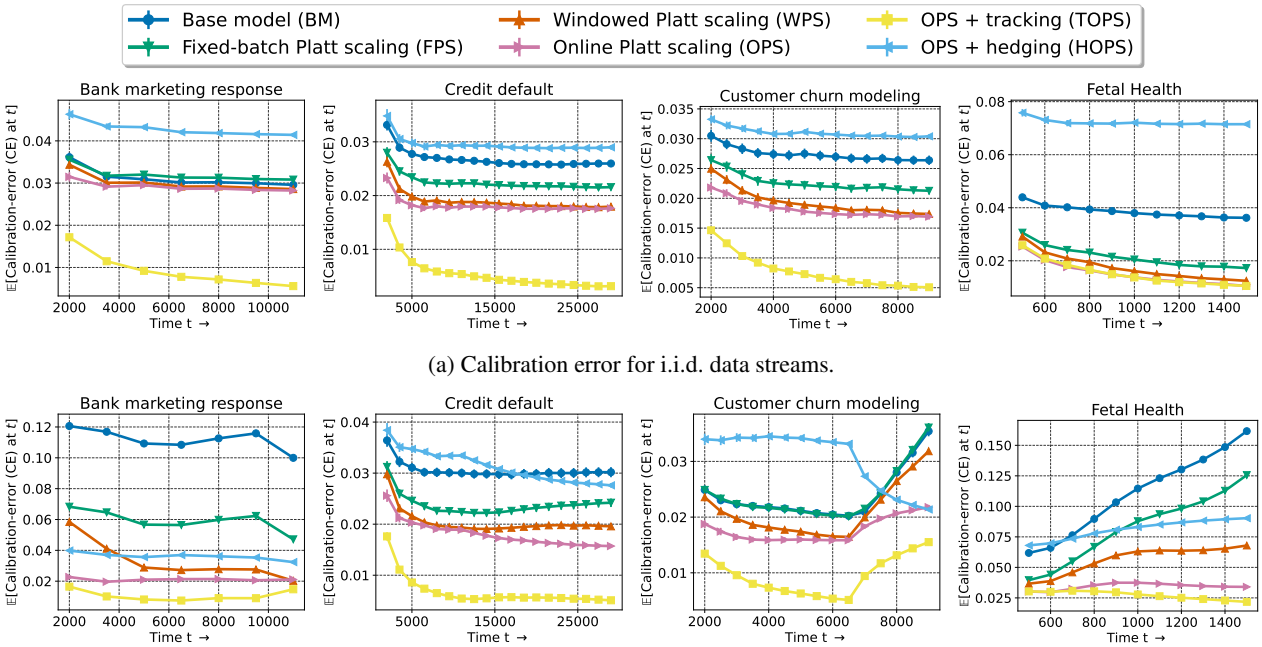
All results in this paper can be reproduced exactly, including the randomization, using the IPython notebooks that can be found at https://github.com/aigen/df-posthoc-calibration in the folder `Online Platt scaling with Calibeating`. The README page in the folder contains a table describing which notebook to run to reproduce individual figures from this paper.



(a) Calibration error for i.i.d. data streams.

(b) Calibration error for drifting data streams.

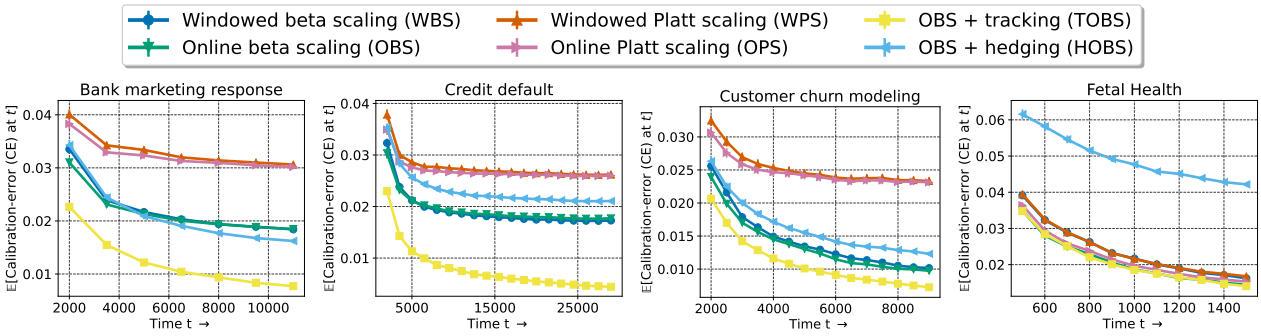
Figure 12: Results for the same experimental setup as Figures 5 and 6, but with $\epsilon = 0.05$.



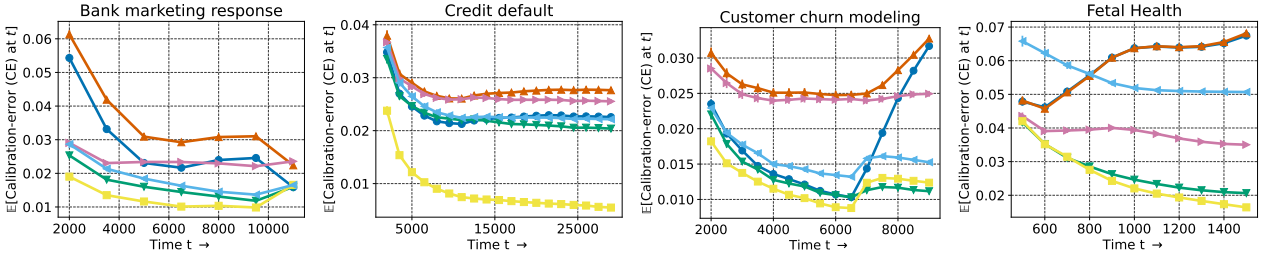
(a) Calibration error for i.i.d. data streams.

(b) Calibration error for drifting data streams.

Figure 13: Results for the same experimental setup as Figures 5 and 6, but with $\epsilon = 0.2$.

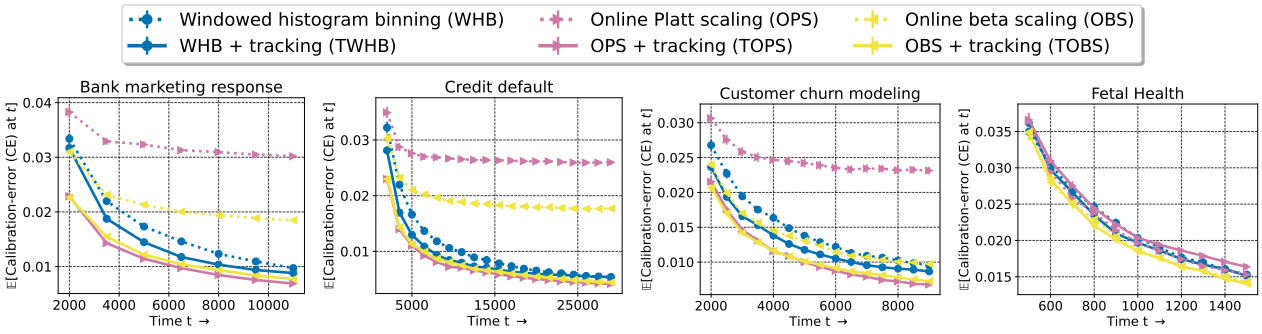


(a) Calibration error for i.i.d. data streams.

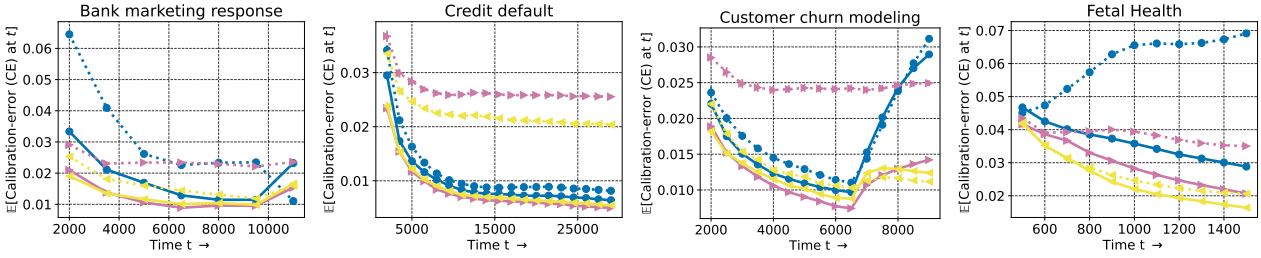


(b) Calibration error for drifting data streams.

Figure 14: Performance of online beta scaling (OBS) and its calibeating variants on real datasets with and without distribution drift. OBS further improves upon OPS in most cases. In each plot, TOBS is the best-performing method.



(a) Calibration error for i.i.d. data streams.



(b) Calibration error for drifting data streams.

Figure 15: Comparing the performance of windowed histogram binning (WHB), online Platt scaling (OPS), online beta scaling (OBS), and their tracking variants on real datasets with and without distribution drifts. Among non-tracking methods (dotted lines), WHB performs well with i.i.d. data, while OBS performs well for drifting data. Among tracking methods (solid lines), TOBS and TOPS are the best-performing methods in every plot. Tracking typically does not improve WHB much since WHB is already a binning method (so tracking is implicit).

Algorithm 1 Online Newton Step for OPS (based on Hazan (2016, Algorithm 12))

Input: $\mathcal{K} = \{(x, y) : \|(x, y)\|_2 \leqslant 100\}$, time horizon T, and initialization parameter $(a_1^{\text{OPS}}, b_1^{\text{OPS}}) = (1, 0) =: \theta_1 \in \mathcal{K}$
Hyperparameters: $\gamma = 0.1$, $\rho = 100$
Set $A_0 = \rho \mathbf{I}_2$
for $t = 1$ **to** T **do**
  Play θ_t, observe log-loss $l(m^{\theta_t}(f(\mathbf{x}_t)), y_t)$ and its gradient $\nabla_t := \nabla_{\theta_t} l(m^{\theta_t}(f(\mathbf{x}_t)), y_t)$
  $A_t = A_{t-1} + \nabla_t \nabla_t^\intercal$
  Newton step: $\widetilde{\theta}_{t+1} = \theta_t - \frac{1}{\gamma} A_t^{-1} \nabla_t$
  Projection: $(a_{t+1}^{\text{OPS}}, b_{t+1}^{\text{OPS}}) = \theta_{t+1} = \arg\min_{\theta \in \mathcal{K}} (\widetilde{\theta}_{t+1} - \theta)^\intercal A_t (\widetilde{\theta}_{t+1} - \theta)$
end for

B Online beta scaling

This is an extended version of Section 5, with some repetition but more details. A recalibration method closely related to Platt scaling is beta scaling (Kull et al., 2017). The beta scaling mapping m has three parameters $(a, b, c) \in \mathbb{R}^3$, and corresponds to a sigmoid transform over two pseudo-features derived from $f(\mathbf{x})$: $\log(f(\mathbf{x}))$ and $\log(1 - f(\mathbf{x}))$:

$$m^{a,b,c}(f(\mathbf{x})) := \text{sigmoid}(a \cdot \log(f(\mathbf{x})) + b \cdot \log(1 - f(\mathbf{x})) + c). \tag{18}$$

Observe that enforcing $b = -a$ recovers Platt scaling since $\text{logit}(z) = \log(z) - \log(1 - z)$. The beta scaling parameters can be learnt following identical protocols as Platt scaling.

- The **traditional method** is to optimize parameters by minimizing the log-likelihood (equivalently, log-loss) over a fixed held-out batch of points.
- A **natural benchmark** for online settings is to update the parameters at some frequency (such as every 50 or 100 steps). At each update, the beta scaling parameters are set to the optimal value based on all data seen so far, and these parameters are used for prediction until the next update occurs. We call this benchmark windowed beta scaling (WBS); it is analogous to the windowed Platt scaling (WPS) benchmark considered in the main paper.
- Our **proposed method** for online settings, called online Beta scaling (OBS), is to use a log-loss regret minimization procedure, similar to OPS. Analogously to (7), R_T for OBS predictions $p_t^{\text{OBS}} = m^{a_t, b_t, c_t}(f(\mathbf{x}_t))$ is defined as

$$R_T(\text{OBS}) = \sum_{t=1}^{T} l(p_t^{\text{OBS}}, y_t) - \min_{(a,b,c) \in \mathcal{B}} \sum_{t=1}^{T} l(m^{a,b,c}(f(\mathbf{x}_t)), y_t), \tag{19}$$

  where $\mathcal{B} := \{(a, b, c) \in \mathbb{R}^3 : a^2 + b^2 + c^2 \leqslant B^2\}$ for some $B \in \mathbb{R}$, and l is the log-loss. We use online Newton step (Algorithm 1) to learn (a_t, b_t, c_t), with the following initialization and hyperparameter values:
  - $\mathcal{K} = \{(x, y, z) : \|(x, y, z)\|_2 \leqslant 100\}$, $(a_1^{\text{OBS}}, b_1^{\text{OBS}}, c_1^{\text{OBS}}) = (1, 1, 0)$;
  - $\gamma = 0.1$, $\rho = 25$, $A_0 = \rho \mathbf{I}_3$.

  These minor changes have to be made simply because the dimensionality changes from two to three. The empirical results we present shortly are based on an implementation with exactly these fixed hyperparameter values that do not change across the experiments (that is, we do not do any hyperparameter tuning).

Due to the additional degree of freedom, beta scaling is more expressive than Platt scaling. In the traditional batch setting, it was demonstrated by Kull et al. (2017) that this expressiveness typically leads to better (out-of-sample) calibration performance. We expect this relationship between Platt scaling and beta scaling to hold for their windowed and online versions as well. We confirm this intuition through an extension of the real dataset experiments of Section 4.1 to include WBS and OBS (Figure 14). In the main paper we reported that the base model (BM) and fixed-batch Platt scaling model (FPS) perform the worst by a margin, so these lines are not reported again. We find that OBS performs better than both OPS and WBS, so we additionally report the performance of calibeating versions of OBS instead of OPS. That is, we replace OPS + tracking (TOPS) with OBS + tracking (TOBS), and OPS + hedging (HOPS) with OBS + hedging (HOBS).

A regret bound similar to Theorem 2.1 can be derived for OBS by instantiating ONS and AIOLI regret bounds with $d = 3$ (instead of $d = 2$ as done for OPS). The calibeating theorems (3.1 and 3.2) hold regardless of the underlying expert, and so also hold for OBS.

C F99 online calibration method

We describe the F99 method proposed by Foster (1999), and used in our implementation of HOPS (Section 3.3). The description is borrowed with some changes from Gupta and Ramdas (2022a). Recall that the F99 forecasts are the mid-points of the ϵ-bins (10): $B_1 = [0, \epsilon), B_2 = [\epsilon, 2\epsilon), \ldots, B_m = [1-\epsilon, 1]$. For $b \in [m] := \{1, 2, \ldots, m\}$ and $t \geqslant 1$, define:

$$\begin{aligned} \text{(mid-point of } B_b) \quad & m_b = (b - 0.5)/m = b\epsilon - \epsilon/2, \\ \text{(left end-point of } B_b) \quad & l_b = (b-1)/m = (b-1)\epsilon, \\ \text{(right end-point of } B_b) \quad & r_b = b/m = b\epsilon, \end{aligned}$$

F99 maintains some quantities as more data set is observed and forecasts are made. These are,

$$\begin{aligned} \text{(frequency of forecasting } m_b) \quad & N_b^t = \left|\{\mathbb{1}\{p_s = m_b\} : s \leqslant t\}\right|, \\ \text{(observed average when } m_b \text{ was forecasted}) \quad & p_b^t = \begin{cases} \sum_{s=1}^t y_s \mathbb{1}\{p_s = m_b\} / N_b^t & \text{if } N_b^t > 0 \\ m_b & \text{if } N_b^t = 0, \end{cases} \\ \text{(deficit)} \quad & d_b^t = l_b - p_b^t, \\ \text{(excess)} \quad & e_b^t = p_b^t - r_b. \end{aligned}$$

The terminology "deficit" is used to indicate that p_b^t is smaller l_b similarly. "Excess" is used to indicate that p_b^t is larger than r_b similarly. The F99 algorithm is as follows. Implicit in the description is computation of the quantities defined above.

F99: the online adversarial calibration method of Foster (1999)

- At time $t = 1$, forecast $p_1 = m_1$.
- At time $t+1$ ($t \geqslant 1$), if

  $$\text{condition A: there exists an } b \in [m] \text{ such that } d_b^t \leqslant 0 \text{ and } e_b^t \leqslant 0,$$

  is satisfied, forecast $p_{t+1} = m_b$ for any i that verifies condition A. Otherwise,

  $$\text{condition B: there exists a } b \in [m-1] \text{ such that } e_b^t > 0 \text{ and } d_{b+1}^t > 0,$$

  must be satisfied (see Lemma 5 (Gupta and Ramdas, 2022a)). For any index b that satisfies condition B, forecast

  $$p_{t+1} = \begin{cases} m_b & \text{with probability } \frac{d_{b+1}^t}{d_{b+1}^t + e_b^t} \\ m_{b+1} & \text{with probability } \frac{e_b^t}{d_{b+1}^t + e_b^t}. \end{cases}$$

  These randomization probabilities are revealed before y_{t+1} is set by the agent that is generating outcomes, but the actual p_t value is drawn after y_{t+1} is revealed.

D Forecasting climatology to achieve calibration

Although Foster and Vohra's result (1998) guarantees that calibrated forecasting is possible against adversarial sequences, this does not immediately imply that the forecasts are useful in practice. To see this, consider an alternating outcome sequence, $y_t = \mathbb{1}\{t \text{ is odd }\}$. The forecast $p_t = \mathbb{1}\{t \text{ is odd }\}$ is calibrated and perfectly accurate. The forecast $p_t = 0.5$ (for every t) is also calibrated, but not very useful.

Thus we need to assess how a forecaster guaranteed to be calibrated for adversarial sequences performs on real-world sequences. In order to do so, we implemented the F99 forecaster (described in Appendix C), on Pittsburgh's hourly rain data from January 1, 2008, to December 31, 2012. The data was obtained from `ncdc.noaa.gov/cdo-web/`. All days on which the hourly precipitation in inches (HPCP) was at least 0.01 were considered as instances of $y_t = 1$. There are many

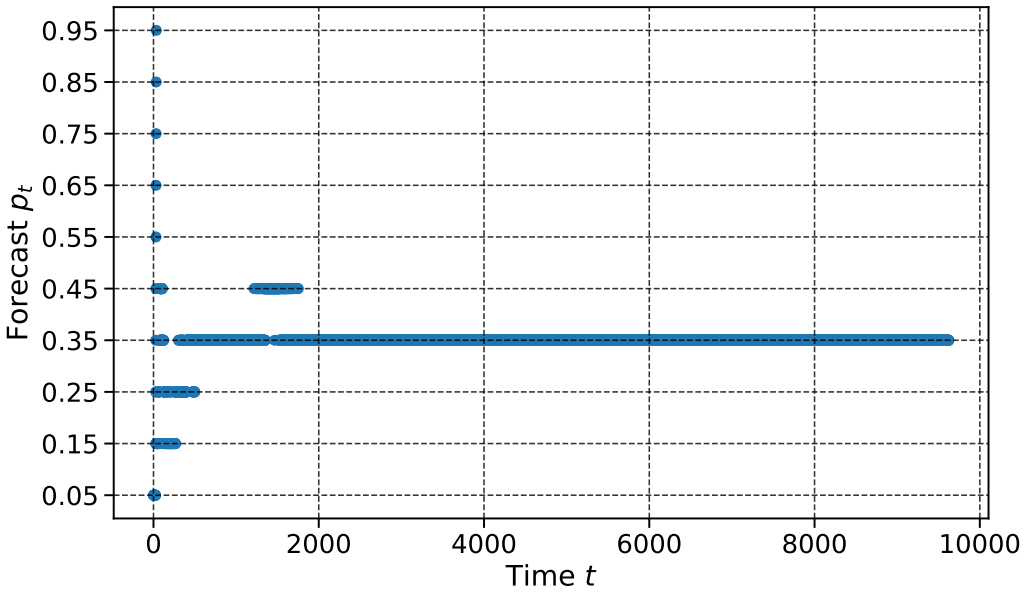


Figure 16: Foster (1999)'s ϵ-calibrated forecaster on Pittsburgh's hourly rain data (2008-2012). The forecaster makes predictions on the grid $(0.05, 0.15, \ldots, 0.95)$. In the long run, the forecaster starts predicting 0.35 for every instance, closely matching the average number of instances on which it rained (≈ 0.37).

missing rows in the data, but no complex data cleaning was performed since we are mainly interested in a simple illustrative simulation. F99 makes forecasts on an ϵ-grid with $\epsilon = 0.1$: that is, the grid corresponds to the points $(0.05, 0.15, \ldots, 0.95)$. We observe (Figure 16) that after around 2000 instances, the forecaster *always* predicts 0.35. This is close to the average number of instances that it did rain which is approximately 0.37 (this long-term average is also called *climatology* in the meteorology literature). Although forecasting climatology can make the forecaster appear calibrated, it is arguably not a useful prediction given that there exist expert rain forecasters who can make sharp predictions for rain that change from day to day.

E Proofs

Proof of Theorem 2.1. The regret bounds for ONS and AIOLI depend on a few problem-dependent parameters.

- The dimension $d = 2$.
- The radius of the reference class B.
- Bound on the norm of the gradient, which for logistic regression is also the radius of the space of input vectors. Due to the assumption on $f(\mathbf{x}_t)$, the norm of the input is at most $\sqrt{\text{logit}(0.01)^2 + 1^2} = \sqrt{\text{logit}(0.99)^2 + 1^2} \leqslant 5$.

The AIOLI bound (9) follows from Theorem 1, equation (4) of Jézéquel et al. (2020), setting $d = 2$ and $R = 10$.

The ONS bound (8) follows from Theorem 4.5 of Hazan (2016), plugging in $G = 5$, $D = 2B$, and $\alpha = e^{-B}$ which is the known exp-concavity constant of the logistic loss over a ball of radius B (Foster et al., 2018).

□

In writing the proofs of the results in Section 3, we will use an object closely connected to sharpness called refinement. For a sequence of forecasts $p_{1:T}$ and outcome sequence $y_{1:T}$, the refinement $\mathcal{R}$ is defined as

$$\mathcal{R}(p_{1:T}) := \frac{1}{T} \sum_{b=1}^{m} N_b \cdot \widehat{y}_b(1 - \widehat{y}_b), \tag{20}$$

where $\widehat{y}_b$ is the average of the outcomes in every ϵ-bin; see the beginning of Section 3.2 where sharpness is defined. The function $x(\in [0,1]) \mapsto x(1-x)$ is minimized at the boundary points $\{0,1\}$ and maximized at $1/2$. Thus refinement is lower if $\widehat{y}_b$ is close to 0 or 1, or in other words if the bins discriminate points well. This is captured formally in the following (well-known) relationship between refinement and sharpness.

Lemma E.1 (Sharpness-refinement lemma)**.** *For any forecast sequence $p_{1:T}$, the refinement $\mathcal{R}$ defined in (20) and the sharpness SHP defined in (12) are related as:*

$$\mathcal{R}(p_{1:T}) = \bar{y}_T - \mathit{SHP}(p_{1:T}),$$

where $\bar{y}_T = \frac{1}{T}\sum_{t=1}^{T} y_t$.

Proof. Observe that

$$\mathcal{R}(p_{1:T}) = \frac{1}{T}\sum_{b=1}^{B} N_b \widehat{y}_b - \frac{1}{T}\sum_{b=1}^{B} N_b \widehat{y}_b^2 = \frac{1}{T}\sum_{b=1}^{B} N_b \widehat{y}_b - \mathrm{SHP}(p_{1:T}).$$

The final result follows simply by noting that

$$\sum_{b=1}^{B} N_b \widehat{y}_b = \sum_{b=1}^{B} \left(\sum_{t \leqslant T, p_t \in B_b} y_t \right) = \sum_{t=1}^{T} y_t.$$

□

We now state a second lemma, that relates $\mathcal{R}$ to the Brier-score $\mathcal{BS}$ defined as

$$\mathcal{BS}(p_{1:T}) := \frac{\sum_{t=1}^{T}(y_t - p_t)^2}{T}. \tag{21}$$

Unlike $\mathcal{R}$ and SHP, $\mathcal{BS}$ is not defined after ϵ-binning. It is well-known (see for example equation (1) of FH23) that if refinement is defined without ϵ-binning (or if the Brier-score is defined with ϵ-binning), then refinement is at most the Brier-score defined above. Since we define $\mathcal{R}$ defined with binning, further work is required to relate the two.

Lemma E.2 (Brier-score-refinement lemma)**.** *For any forecast sequence $p_{1:T}$ and outcome sequence $y_{1:T}$, the refinement $\mathcal{R}$ and the Brier-score $\mathcal{BS}$ are related as*

$$\mathcal{R}(p_{1:T}) \leqslant \mathcal{BS}(p_{1:T}) + \frac{\epsilon^2}{4} + \epsilon, \tag{22}$$

where ϵ is the width of the bins used to define $\mathcal{R}$ (10).

Proof. Define the discretization function $\mathrm{disc} : [0,1] \to [0,1]$ as $\mathrm{disc}(p) = \text{mid-point}(B_b) \iff p \in B_b$. Note that for all $p \in [0,1]$, $|p - \mathrm{disc}(p)| \leqslant \epsilon/2$. Based on standard decompositions (such as equation (1) of FH23), we know that

$$\mathcal{R}(p_{1:T}) \leqslant \frac{\sum_{t=1}^{T}(y_t - \mathrm{disc}(p_t^{\mathrm{TOPS}}))^2}{T}. \tag{23}$$

We now relate the RHS of the above equation to $\mathcal{BS}$

$$\begin{aligned}
\sum_{t=1}^{T}(y_t - \mathrm{disc}(p_t))^2 &= \sum_{t=1}^{T}(y_t - p_t + p_t - \mathrm{disc}(p_t))^2 \\
&= T \cdot \mathcal{BS}(p_{1:T}) + \sum_{t=1}^{T}(p_t - \mathrm{disc}(p_t))^2 + 2\sum_{t=1}^{T}(y_t - p_t)(p_t - \mathrm{disc}(p_t)) \\
&\leqslant T \cdot \mathcal{BS}(p_{1:T}) + T(\epsilon/2)^2 + 2\sum_{t=1}^{T}|y_t - p_t|\,(\epsilon/2). \\
&\leqslant T \cdot \mathcal{BS}(p_{1:T}) + T(\epsilon/2)^2 + T\epsilon.
\end{aligned}$$

The result of the theorem follows by dividing by T on both sides. □

Proof of Theorem 3.1. The calibeating paper (Foster and Hart, 2023) is referred to as FH23 in this proof for succinctness.

We use Theorem 3 of FH23, specifically equation (13), which gives an upper bound on the Brier-score of a tracking forecast ($\mathcal{B}_t^c$ in their notation) relative to the refinement (20) of the base forecast. In our case, the tracking forecast is TOPS, the base forecast is OPS, and FH23's result gives,

$$\mathcal{BS}(p_{1:T}^{\text{TOPS}}) = \frac{\sum_{t=1}^{T}(y_t - p_t^{\text{TOPS}})^2}{T} \leqslant \mathcal{R}(p_{1:T}^{\text{TOPS}}) + \frac{\log T + 1}{\epsilon T}. \tag{24}$$

Using the Brier-score-refinement lemma E.2 to lower bound $\mathcal{BS}(p_{1:T}^{\text{TOPS}})$ gives

$$\mathcal{R}(p_{1:T}^{\text{TOPS}}) - \frac{\epsilon^2}{4} - \epsilon \leqslant \mathcal{R}(p_{1:T}^{\text{OPS}}) + \frac{\log T + 1}{\epsilon T}. \tag{25}$$

Finally, using the sharpness-refinement lemma E.1, we can replace each $\mathcal{R}$ with $\bar{y}_T - \text{SHP}$. Rearranging terms gives the final bound.

□

Proof of Theorem 3.2. The calibeating paper (Foster and Hart, 2023) is referred to as FH23 in this proof for succinctness.

Sharpness bound (16). Theorem 5 of FH23 showed that the expected Brier-score for a different hedging scheme (instead of F99), is at most the expected refinement score of the base forecast plus $\epsilon^2 + \frac{\log T+1}{\epsilon^2 T}$. In our case, the second term remains unchanged, but because we use F99, the ϵ^2 needs to be replaced, and we show that it can be replaced by ϵ next.

Let us call the combination of OPS and the FH23 hedging method as FH23-HOPS, and the calibeating forecast as $p_{1:T}^{\text{FH23-HOPS}}$. The source of the ϵ^2 term in Theorem 5 of FH23 is the following property of FH23-HOPS: for both values of $y_t \in \{0, 1\}$,

$$\mathbb{E}_{t-1}\left[(y_t - p_t^{\text{FH23-HOPS}})^2 - (y_t - \text{Average}\{y_s : s < t, p_s^{\text{OPS}} = p_t^{\text{OPS}}, p_s^{\text{FH23-HOPS}} = p_t^{\text{FH23-HOPS}}\})^2\right] \leqslant \epsilon^2,$$

where $\mathbb{E}_{t-1}[\cdot]$ is the expectation conditional on $(y_{1:t-1}, p_{1:t-1}^{\text{FH23-hedging}}, p_{1:t-1}^{\text{OPS}})$ (all that's happened in the past, and the current OPS forecast). For HOPS, we will show that

$$Q_t := \mathbb{E}_{t-1}\left[(y_t - p_t^{\text{HOPS}})^2 - (y_t - \text{Average}\{y_s : s < t, p_s^{\text{OPS}} = p_t^{\text{OPS}}, p_s^{\text{HOPS}} = p_t^{\text{HOPS}}\})^2\right] \leqslant \epsilon,$$

for $y_t \in \{0, 1\}$, which would give the required result.

At time t, the F99 forecast falls into one of two scenarios which we analyze separately (see Appendix C for details of F99 which would help follow the case-work).

- **Case 1.** This corresponds to condition A in the description of F99 in Section C. There exists a bin index b such that $q = \text{mid-point}(B_b)$ satisfies

  $$\left|\text{Average}\{y_s : s < t, p_s^{\text{OPS}} = p_t^{\text{OPS}}, p_s^{\text{HOPS}} = q\} - q\right| \leqslant \epsilon/2.$$

  In this case, F99 would set $p_t^{\text{HOPS}} = q$ (deterministically) for some q satisfying the above. Thus,

  $$\begin{aligned} Q_t &= (y_t - q)^2 - (y_t - \text{Average}\{y_s : s < t, p_s^{\text{OPS}} = p_t^{\text{OPS}}, p_s^{\text{HOPS}} = q\})^2 \\ &\leqslant \max((y_t - q)^2 - (y_t - q - \epsilon/2)^2, (y_t - q)^2 - (y_t - q + \epsilon/2)^2) \\ &\leqslant (\epsilon/2)(2\,|y_t - q| + \epsilon/2) < \epsilon, \end{aligned}$$

  irrespective of y_t, since $q \in [\epsilon/2, 1 - \epsilon/2]$.

- **Case 2.** This corresponds to condition B in the description of F99 in Section C. If Case 1 does not hold, F99 randomizes between two consecutive bin mid-points $m - \epsilon/2$ and $m - \epsilon/2$, where m is one of the edges of the ϵ-bins (10). Define $n_1 := \text{Average}\{y_s : s < t, p_s^{\text{OPS}} = p_t^{\text{OPS}}, p_s^{\text{HOPS}} = m - \epsilon/2\}$ and $n_2 := \text{Average}\{y_s : s < t, p_s^{\text{OPS}} = p_t^{\text{OPS}}, p_s^{\text{HOPS}} = m + \epsilon/2\}$. The choice of m in F99 guarantees that $n_2 < m < n_1$, and the randomization probabilities are given by

  $$\mathbb{P}_{t-1}(p_t^{\text{HOPS}} = m - \epsilon/2) = \frac{m - n_2}{n_1 - n_2}, \text{ and } \mathbb{P}_{t-1}(p_t^{\text{HOPS}} = m + \epsilon/2) = \frac{n_1 - m}{n_1 - n_2},$$

where $\mathbb{P}_{t-1}$ is the conditional probability in the same sense as $\mathbb{E}_{t-1}$. We now bound Q_t. If $y_t = 1$,

$$\begin{aligned}
Q_t &= \mathbb{E}_{t-1}\left[(y_t - p_t^{\text{HOPS}})^2 - (y_t - \text{Average}\{y_s : s < t, p_s^{\text{OPS}} = p_t^{\text{OPS}}, p_s^{\text{HOPS}} = p_t^{\text{HOPS}}\})^2\right] \\
&= \frac{m - n_2}{n_1 - n_2}\left((1-(m-\epsilon/2))^2 - (1-n_1)^2\right) + \frac{n_1 - m}{n_1 - n_2}\left((1-(m+\epsilon/2))^2 - (1-n_2)^2\right) \\
&= \underbrace{\frac{m - n_2}{n_1 - n_2}\left((1-m)^2 - (1-n_1)^2\right) + \frac{n_1 - m}{n_1 - n_2}\left((1-m)^2 - (1-n_2)^2\right)}_{=:A_1} \\
&\quad + \underbrace{2\cdot(\epsilon/2)\cdot\frac{(m-n_2)(1-m) - (n_1-m)(1-m)}{n_1-n_2}}_{=:A_2} \\
&\quad + (\epsilon/2)^2 \cdot \frac{n_1 - n_2}{n_1 - n_2}.
\end{aligned}$$

A_1 and A_2 simplify as follows.

$$\begin{aligned}
A_1 &= \frac{(m-n_2)(n_1-m)(2-(n_1+m)) + (n_1-m)(n_2-m)(2-(n_2+m))}{n_1-n_2} \\
&= \frac{(m-n_2)(n_1-m)(n_2-n_1)}{n_1-n_2} < 0,
\end{aligned}$$

since $n_2 < m < n_1$.

$$\begin{aligned}
A_2 &= \epsilon\cdot\frac{(m-n_2)(1-m)}{n_1-n_2} + \epsilon\cdot\frac{(m-n_1)(1-m)}{n_1-n_2} \\
&< \epsilon\cdot\frac{(m-n_2)(1-m)}{n_1-n_2} && \text{(since } m < n_1\text{)} \\
&< \epsilon(1-m).
\end{aligned}$$

Overall, we obtain that for $y_t = 1$,

$$Q_t < \epsilon(1-m) + (\epsilon^2/4) < \epsilon,$$

where the final inequality holds since m is an end-point between two bins, and thus $m \geqslant \epsilon$. We do the calculations for $y_t = 0$ less explicitly since it essentially follows the same steps:

$$\begin{aligned}
Q_t &= \mathbb{E}_{t-1}\left[(0 - p_t^{\text{HOPS}})^2 - (0 - \text{Average}\{y_s : s < t, p_s^{\text{OPS}} = p_t^{\text{OPS}}, p_s^{\text{HOPS}} = p_t^{\text{HOPS}}\})^2\right] \\
&= \frac{m-n_2}{n_1-n_2}\left((m-\epsilon/2)^2 - n_1^2\right) + \frac{n_1-m}{n_1-n_2}\left((m+\epsilon/2)^2 - n_2^2\right) \\
&= \frac{(m-n_2)(m-n_1)(m+n_1) + (n_1-m)(m-n_2)(m+n_2)}{n_1-n_2} + \epsilon\cdot\frac{(n_2-m)m + (n_1-m)m}{n_1-n_2} + \frac{\epsilon^2}{4} \\
&< 0 + \epsilon m + (\epsilon^2/4) < \epsilon.
\end{aligned}$$

Finally, by Proposition 1 of FH23 and the above bound on Q_t, we obtain,

$$\mathbb{E}\left[\mathcal{R}(p_{1:T}^{\text{HOPS}})\right] \leqslant \mathbb{E}\left[\mathcal{BS}(p_{1:T}^{\text{HOPS}})\right] \leqslant \epsilon + \mathcal{R}(p_{1:T}^{\text{OPS}}) + \frac{\log T + 1}{\epsilon^2 T}.$$

Using the sharpness-refinement lemma E.1, we replace each $\mathcal{R}$ with $\bar{y}_T - \text{SHP}$. Rearranging terms gives the sharpness result.

Calibration bound (17)**.** Recall that the number of bins is $m = 1/\epsilon$. For some bin indices $b, b' \in \{1, 2, \ldots, m\}$, let $S_{b\to b'} = \{t \leqslant T : p_t^{\text{OPS}} \in B_b, p_t^{\text{HOPS}} = \text{mid-point}(B_{b'})\}$ be the set of time instances at which the OPS forecast p_t^{OPS} belonged to bin b, but the HOPS forecast p_t^{HOPS} belonged to bin b' (and equals the mid-point of bin b'). Also, let $S_b = \{t \leqslant T : p_t^{\text{OPS}} \in B_b\}$ be the set of time instances at which the p_t^{OPS} forecast belonged to bin b. Thus $S_b = \bigcup_{b'=1}^{m} S_{b\to b'}$. Also define $N_b^{\text{OPS}} = |S_b|$ and $N_b^{\text{HOPS}} = \left|\{t \leqslant T : p_t^{\text{HOPS}} = \text{mid-point}(B_b)\}\right|$.

Now for any specific b, consider the sequence $(y_t)_{t \in S_b}$. On this sequence, the HOPS forecasts correspond to F99 using just the outcomes (with no regard for covariate values once the bin of p_t^{OPS} is fixed). Thus, within this particular bin, we have a usual CE guarantee that F99's algorithm has for any arbitrary sequence:

$$\underbrace{\mathbb{E}\left[\frac{1}{N_b^{\text{OPS}}}\sum_{b'=1}^{m}\left|\sum_{t \in S_{b \to b'}}(y_t - p_t^{\text{HOPS}})\right|\right]}_{\text{this is the expected CE over the } S_b \text{ instances}} \leqslant \frac{\epsilon}{2} + \frac{2}{\sqrt{\epsilon \cdot N_b^{\text{OPS}}}}. \tag{26}$$

This result is unavailable in exactly this form in Foster (1999) which just gives the reduction to Blackwell approachability, after which any finite-sample approachability bound can be used. The above version follows from Theorem 1.1 of Perchet (2014). The precise details of the Blackwell approachability set, reward vectors, and how the distance to the set can be translated to CE can be found in Gupta and Ramdas (2022a, Section 4.1).

Jensen's inequality can be used to lift this CE guarantee to the entire sequence:

$$\begin{aligned}
\mathbb{E}\left[\text{CE}(p_{1:T}^{\text{HOPS}})\right] &= \mathbb{E}\left[\sum_{b=1}^{m}\frac{\left|N_b^{\text{HOPS}}(\widehat{y}_b^{\text{HOPS}} - \widehat{p}_b^{\text{HOPS}})\right|}{T}\right] \\
&= \mathbb{E}\left[\sum_{b=1}^{m}\frac{\left|\sum_{t=1}^{T}(y_t - p_t^{\text{HOPS}})\mathbb{1}\left\{p_t^{\text{HOPS}} \in B_b\right\}\right|}{T}\right] \\
&= \mathbb{E}\left[\sum_{b=1}^{m}\frac{\left|\sum_{b'=1}^{m}\sum_{t \in S_{b' \to b}}(y_t - p_t^{\text{HOPS}})\right|}{T}\right] \\
&\leqslant \mathbb{E}\left[\sum_{b=1}^{m}\sum_{b'=1}^{m}\frac{\left|\sum_{t \in S_{b' \to b}}(y_t - p_t^{\text{HOPS}})\right|}{T}\right] \qquad \text{(Jensen's inequality)} \\
&= \sum_{b'=1}^{m}\mathbb{E}\left[\sum_{b=1}^{m}\frac{\left|\sum_{t \in S_{b' \to b}}(y_t - p_t^{\text{HOPS}})\right|}{T}\right] \\
&\leqslant \sum_{b'=1}^{m}\frac{N_{b'}^{\text{OPS}}\left(\epsilon/2 + 2/\sqrt{\epsilon \cdot N_{b'}^{\text{OPS}}}\right)}{T} \qquad \text{(by (26))} \\
&= \frac{\epsilon}{2} + \frac{2}{\sqrt{\epsilon}} \cdot \frac{\sum_{b'=1}^{m}\sqrt{N_{b'}^{\text{OPS}}}}{\sum_{b'=1}^{m}N_{b'}^{\text{OPS}}} \qquad \left(\text{since } T = \sum_{b'=1}^{B}N_{b'}^{\text{OPS}}\right) \\
&\overset{(\star)}{\leqslant} \frac{\epsilon}{2} + \frac{2}{\sqrt{\epsilon}} \cdot \sqrt{\frac{m}{T}} = \frac{\epsilon}{2} + 2\sqrt{\frac{1}{\epsilon^2 T}},
\end{aligned}$$

as needed to be shown. The inequality $(\star)$ holds because, by Jensen's inequality (or AM-QM inequality),

$$\sqrt{\frac{\sum_{b'=1}^{m}N_{b'}^{\text{OPS}}}{m}} \geqslant \frac{\sum_{b'=1}^{m}\sqrt{N_{b'}^{\text{OPS}}}}{m},$$

so that

$$\frac{\sum_{b'=1}^{m}\sqrt{N_{b'}^{\text{OPS}}}}{\sum_{b'=1}^{m}N_{b'}^{\text{OPS}}} = \frac{\sum_{b'=1}^{m}\sqrt{N_{b'}^{\text{OPS}}}}{\sqrt{\sum_{b'=1}^{m}N_{b'}^{\text{OPS}}}} \cdot \frac{1}{\sqrt{\sum_{b'=1}^{m}N_{b'}^{\text{OPS}}}} \leqslant \frac{\sqrt{m}}{\sqrt{\sum_{b'=1}^{m}N_{b'}^{\text{OPS}}}} = \sqrt{m/T}.$$

□